Besök på General Electric CRD, 20-23 mars 1984

Åström, Karl Johan; Andersson, Leif

1985

Document Version:
Publisher's PDF, also known as Version of record

Link to publication

Citation for published version (APA):
Åström, K. J., & Andersson, L. (1985). *Besök på General Electric CRD, 20-23 mars 1984*. (Travel Reports TFRT-8042). Department of Automatic Control, Lund Institute of Technology (LTH).

Total number of authors:
2

LUND UNIVERSITY

PO Box 117
221 00 Lund
+46 46-222 00 00

CODEN: LUTFD2/(TFRT-8042)/1-84(1985)

BESÖK PÅ GENERAL ELECTRIC CRD

20 - 23 MARS 1984

K J ÅSTRÖM OCH LEIF ANDERSSON

INSTITUTIONEN FÖR REGLERTEKNIK
LUNDS TEKNISKA HÖGSKOLA
SEPTEMBER 1985

LUND INSTITUTE OF TECHNOLOGY
DEPARTMENT OF AUTOMATIC CONTROL
Box 725
S 220 07 Lund 7 Sweden

Document name
Report

Date of issue
September 1985

Document number
CODEN: LUTFD2/(TFRT-8042)/1-84/(1985)

Supervisor

Sponsoring organization

Author(s)
K J Åström
Leif Andersson

Title and subtitle
Besök på General Electric CRD , 20-23 mars 1984. (Visit at General Electric CRD, March 20-23, 1984).

Abstract
This travel report describes a visit at General Electric's central research laboratory. The purpose has been to take part of GE:s experiences of using our program packages Simnon and Idpac.

Key words

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
Swedish

Number of pages
84

Recipient's notes

Security classification

BESÖK PÅ GENERAL ELECTRIC CRD

20 - 23 mars 1984

K J Åström och Leif Andersson

SAMMANFATTNING

Denna rapport redogör för ett besök på General Electrics centrala forsknings-laboratorium för att ta del av GE:s erfarenheter av våra programpaket Simnon och Idpac.

INNEHÅLL

1. Inledning

Den centrala forsknings- och utvecklingsavdelningen (CRD) vid General Electric i Schenectady förvärvade våra programpaket Simnon och Idpac i källkod 1981. Programmen har av CRD anpassats och distribuerats till olika divisioner på GE. Representanter för GE har också vid olika tillfällen besökt institutionen för att diskutera paketen och deras användning. Inom GE hade man nu en omfattande erfarenhet av användning av våra och andras paket för CACSD (Computer Aided Control System Design). Avsikten med besöket var att göra en samlad genomgång av erfarenheter med programvaran bl.a. med tanke på det planerade CACE projektet.

Agendan för vårt besök framgår av Appendix A. I avsnitt 2 ges en sammanfattande diskussion av GE:s erfarenheter av våra paket. Avsnitt 3 beskriver det s.k. "federated system" som GE har byggt upp kring våra och andras paket. Några av GE:s framtida planer för CACSD beskrivs i avsnitt 4. Vi diskuterade också några reglertekniska spörsmål utöver paketen. En sammanfattande diskussion av detta ges i avsnitt 5. En sammanfattning av slutsatsen från besöket ges i avsnitt 6. Kopior på stordia och annat material från olika presentationer ges i Appendices.

2. Erfarenheter av Simnon och Idpac

GE förvärvade programpaketen Idpac och Simnon i källkodsform 1981. Programmen har inlemmats i ett s.k. "federated system". Där finns också CLADP, ett paket för flervariabel syntes från England och ett paket för analys och dimensionering av system på tillståndsform. Det "federerade systemet" var uppbyggt med hjälp av Intrac med tillägg av några kommandon. Programutvecklingen utfördes av den centrala forsknings- och utvecklingsavdelningen (CRD). Programmen distribuerades sedan ut till de olika operativa avdelningarna vid GE. Den utvecklingsinsats om gjorts har väsentligen bestått av systemintegration, dokumentation och utbildning. Speciella simnonsystem, fortransystem och macron hade också tagits fram i samband med spridningen. Programmen används nu rutinmässigt på den centrala utvecklingsavdelningen och på ca. 10

andra avdelningar inom GE.

Erfarenheterna från användning av paketen var genomgående goda och positiva. Simnon och Idpac nyttjades i ungefär samma omfattning. Det var också intressant att notera att det "federerade systemet" baserades på Intrac. Vi fick också det trevliga beskedet att programvarukvaliteten var avsevärt högre på paketen från Lund än t.ex. paketen från England. På följande sidor ges en sammafattning av olika tillämpningar och en bedömning av paketen Simnon och Idpac.

GE EXPERIENCE - CACSD

BASICALLY QUITE FAVORABLE.

10 DEPARTMENTS:

- ARMAMENT SYSTEMS DEPARTMENT - GUN TURRETS
- ORDNANCE SYSTEMS DEPARTMENT - POLARIS SUB. TANK TRANSMISSIONS
- REENTRY SYSTEMS OPERATION - REENTRY VEHICLES
- SPACE DIVISION - COMMUNICATIONS SATELLITE
- AIRCRAFT EQUIPMENT DEPARTMENT - AUTOPILOTS
- AIRCRAFT ENGINE GROUP
  - EVENDALE - JET ENGINES
  - LYNN - HELICOPTER ENGINES
- ELECTRIC UTILITY SYSTEMS ENG. DEPARTMENT - ELECTRIC UTILITY STUDIES
- GAS TURBINE DEPARTMENT - GAS TURBINE CONTROLS
- ROBOTICS AND VISION SYSTEMS DEPARTMENT - ROBOTS

PACKAGES	DELIVERED TO USERS
CLADP	1981
SIMNON	1982
IDPAC	1982
SSDP	1983

IDPAC

BASIC EVALUATION: EXCELLENT

USES:

- IGCC
- COMPRESSOR STALL
- ELECTRIC UTILITY

LIMITATIONS:

- DISCRETE VERSUS CONTINUOUS

SIMNON

BASIC EVALUATION: EXCELLENT

FAST RESPONSE, SMALL TO MEDIUM PROBLEMS

USES: COMMUNICATION SATELLITE

JET ENGINE ACTUATORS

MULTI-ENGINE FICA

ROBOTICS

MCS REACTOR

LIMITATIONS: RUNNING TIME/CPU LOAD

FASTER HELP MESSAGES

LANGUAGE

- MATRIX (SHOULD BE MATLAB LIKE)
- DO LOOPS
- GO TO

ALGORITHMS: NEW INTEGRATION ROUTINES

- EULER, JPL

DISCONTINUOUS NON-LINEARITIES

GRAPHICS

NITTY-GRITTY: CONTINUATION OF INPUT LINES

>= <= INEQUALITIES

NUMBER OF ELEMENTS IN LOGGER FILE

Utbildning

GE har lagt ned möda på att utbilda representanter för olika divisioner i användning av programpaketen. I utbildningsmomenten har man samarbetat med RPI (Rensselaer Polytechnic Institute). På detta sätt har man också på RPI snabbt fått tillgång till nya kurser och ny kunskap. I Appendix B ges kursplaner för två kurser

Computer-Aided System Modeling and Control
System Identification and Self-tuning Control

som utvecklats på detta sätt. Vid diskussionen redogjorde vi för de externkurser som utvecklats vid LTH.

Sammanfattningsvis kunde det konstateras att det tog lång tid och stora resurser att genomföra utbildningen. Det gäller dels utbildning i reglerteknisk metodik och dels utbildning i användning av paketen. Användare vill ha mycket av båda. En bra modell har varit att se till att det på varje avdelning finns åtminstone en kunnig person som kan sprida kunskapen lokalt.

EDUCATION

9 MANUALS FOR CLADP, SIMNON, IDPAC

- TUTORIAL/MACRO FILES
- REFERENCE MANUALS

2 COURSES GIVEN TO BEGIN INTRODUCTION

- CLADP MULTIVARIABLE
- SIMNON / IDPAC

NEEDS:

- SSDP MANUALS
- CLADP REFERENCE MANUAL
- MULTIVARIABLE COURSE NOTES/BOOK

Speciella problem

Vi diskuterade speciella problem med olika personer som använt paketen. Bland sådana specialproblem fanns: simulering av glapp och friktion, organisering av modellbibliotek.

3. Det Federerade Systemet

Som nämndes i avsnitt 2 har man på GE varit intresserad av att köra flera olika programpaket i en gemensam miljö. Till att börja med det engelska paketet CLADP och våra paket Idpac och Simnon. På GE hade man funnit Intrac vara ett utmärkt hjälpmedel för att integrera olika programpaket. Intrac kan ju ses som ett hjälpmedel för att göra en uppsättning Fortran-subrutiner till ett interaktivt system. Projektet är intressant för det visar att programpaket av olika ursprung kan fogas samman till en enhet med måttlig ansträngning. En beskrivning av systemet ges i Appendix C. Programmet fungerar så att paket som Idpac och Simnon kan köras som tidigare oberoende av varandra men att det också går lätt att kommunicera mellan olika paket.

4. GE:s CACSD Planer

Erfarenheterna av datorstödd konstruktion av reglersystem har varit mycket positiva. Framförallt har man fått en drastisk reduktion av den mantid som gått åt för att genomföra projekt. Typiska siffror var reduktion i mantid med en faktor fyra och en motsvarande reduktion i total projekttid. Projekt som tidigare gjorts på 4 månader kunde nu göras på 1 månad. Dessa siffror stämmer väl med våra egna erfarenheter. Man såg därför CACSD som en viktig produktionshöjande faktor.

Mindre projekt

I de framtida planerna för CACSD fanns dels marginella utvidgningar av existerande paket. I denna kategori finns t.ex. Simnon kommandon för beräkning av stationära värden och linjärisering. Se Appendix D.

Större projekt

Bland mer ambitiösa projekt fanns ett projekt för att införa avancerad filtrering och parameterskattning i Simnon. Detta projekt drevs på ett intressant sätt. Man hade bildat en grupp av GE-ingenjörer från många olika avdelningar, Re-entry systems operation, military electronics, corporate research and development, space systems operation, electronics lab. aerospace electronics and control, ordnance systems. Denna grupp skulle tjäna som en länk mellan de olika avdelningarna och som en länk med omvärlden för att diskutera filtreringsproblem, kanalisera erfarenheter och utveckla projekt av gemensamt intresse för de olika avdelningarna. Gruppen hade definierat en utvidgning av Simnon för att lösa och simulera filtrerings- och skattningsproblem. En sammanfattning av gruppens aktiviteter ges i Appendix E.

Grafik

Man hade också funnit ett stort intresse för att arbeta med grafik. Några synpunkter sammanfattas på stordian på nästa sida. Vi diskuterade också konsekvenser av att arbetsstationer blir tillgängliga i större omfattning.

GRAPHICS

USERS WANT WIDE RANGE OF DISPLAYS

- QUICK DISPLAYS
- PUBLICATION QUALITY
- MORE CONTROL OF DISPLAY

NEED:

- COLOR
- 'MATLAB' FOR GRAPHICS

BASIC COMMENTS:

WE PRESENT CONTROL DESIGN WITH SAME WAY AS DONE IN 1935.

WHAT PROMISE WORK STATIONS?

Användning av expertsystem

Man har experimenterat med att använda expertsystem för dimensionering av reglersystem. Ett experimentsystem som baserats på GE:s system DELTA (Diesel-Electric Locomotive Troubleshooting Aid) hade utvecklats för syntes av fasavancerande kompenseringsnät. Se appendix F.

En vidare ram

Vi förde också en diskussion som betonade att problemet att dimensionera ett reglersystem ses i ett vidare sammanhang. Siffror från flera olika projekt visade att regulatordimensionering ofta utgör mindre än 20% av totala projekten. Modellbygge och parameterskattning är ofta mer omfattande. Ur denna synpunkt har vårt CACE-projekt rätt inriktning ty i detta finns det stort utrymme för modellbygge, identifiering och simulering. Möjligheterna att integrera implementeringsdelen diskuterades också. Man skulle t.ex. kunna tänka sig att ha programmoduler för kodgenerering för specialprocessorer anslutna till syntes-paketen.

5. Övrigt

Flera andra reglertekniska problem diskuterades också. Se Appendix A. Bl.a. gav jag en översiktlig presentation av forskning och undervisning i reglerteknik vid LTH. Speciellt presenterade jag våra ideer om CACE projektet. Vi fick också se flera av laboratorierna, bl.a. ett mycket intressant projekt där en svetsrobot styrdes med ett bildbaserat system. Bland de frågor som diskuterades märktes

Adaptiv reglering
Automatinställning
Olinjära system
Reglerproblem för industrirobotar
Reglering av system med mekaniska resonanser
Skräddarsydda signalprocessorer för reglering

Blandning av reglering och logisk styrning

Realtidsprogrammering

Integration av bildbehandling och reglering

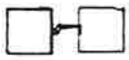
Bland de mer intressanta projekten var en speciell VLSI-krets för reglering och logisk styrning. Kretsen som kallas CPAC (Concurrent Processor Architecture for Control) har två processorelement, ett som tar hand om reglerfunktioner CPE (Continuous Processing Element) och ett som tar hand om logikstyrning och sekvensering DPE (Discrete Processing Element).

6. Slutsatser

Besöket var mycket givande. Vi fick en god inblick i hur man på GE använder våra programpaket. Vi fick också goda uppslag för fortsatt arbete på kort och lång sikt. Vi fick också styrkt att våra planer på det nya CACE projektet är riktiga. Vi kom också överens om att fortsätta vårt samarbete med utbyte av idéer och erfarenheter. Stordiabilderna på nästa sidor sammanfattar de områden där ett fortsatt samarbete bedöms intressant.

- CACSD
    - CONTINUED EXCHANGE
    - EFFECTIVE USE OF EXISTING TOOLS
    - SHARE EDUCATIONAL EXPERIENCES
    - MACRO LIBRARIES
    - EXPERT SYSTEM - IDPAC

- RESONANCE/ELASTIC MODES PROBLEM

 - LAB

- NONLINEAR SYSTEMS
    - LOGIC + CONTROL

- SOFTWARE/REAL TIME PROGRAM
    - LICS - GRAPHIC FRONT ENDS

- ONLINE "SMART" DIAGNOSTICS
    - AI/EXPERT SYSTEMS
    - WATCH AND EVALUATE OPERATION

FUTURES

- POORLY MODELED PROCESSES

  EMULATE "HUMAN OPERATOR"

- DISTRIBUTED COMPUTER CONTROLS

- INTEGRATION WITH VISION AND CONTROL

- CACSD ⟶ IMPLEMENTATION

Appendix A

Agenda for Lund Visit

1. Control Theory Branch Overview (Cassidy)
2. Lund Overview (Astrom)
3. Computer Aided Design for Control
    a) GE Experience with CACSD especially Lund packages (Spang)
    b) GE CACSD
        Federated System (discussion and demo) (Spang)
        Expert Systems (Taylor)
    c) Lund update and discussion (Astrom)
    d) Control Engineer's Workbench (Sutherland)
        Hierarchical Control System Emulator
        Sequential Translator
        Software Engineering Tools
    e) Future Directions
        Expert Systems
        Implementation Tools
        ??????
4. Discussion on Future of Control. Is Control Dead?
    Where is it going? What are the issues?
5. Review of Robotics Projects (Sweet)
6. Hardware Architectures for Control
    Concurrent Processor Architecture for Control (CPAC)
7. Lund Projects
    Adaptive Control
    Picture Processing
    Other

Appendix B

COMPUTER-AIDED SYSTEM MODELING AND CONTROL

The course will be directed towards the study of interactive computer-aided analysis and design algorithms for dynamic system modeling and control.Many practical aspects of experimental data analysis and computer-aided control system design will be discussed.The published literature representing applications of computer-aided model development and control in aerospace and chemical processes will be reviewed. The course will include "hands-on" experience with the identification software "IDPAC"and the nonlinear simulation software "SIMNON" developed at Lund Institute of Technology in Sweden.

The course will require a control design project for each participant.

The course description is summerized in the following.

1 Basic Approach to Parameter Estimation

- -Least squares algorithm
- -General characteristics of estimators
- -Mean square and minimum variance estimator (3 WEEKS)
- -Principle of maximum likelihood

2 Introduction to Identification of Dynamic Models

- -Autoregressive and moving average Models
- -Method of least-squares
- -Problem of correlated residuals and biased estimates (2 WEEKS)
- -maximum likelihood algorithm

3 Principles of IDPAC Software

- -Interactive structure of IDPAC
- -Review of IDPAC commands
- -Examples of model identification with maximum likelihood
- -Comparison with least-squares models
- -Computation of system frequency response
- -Autocorrelation function and power spectral density (3 WEEKS)
- -Statistical tests of residuals
- -Practical aspects of Identification

4 Introduction to SIMNON

- -Interaction principles
- -Discrete and continuous system simulation
- \- Simulation of a computer control system

(2 WEEKS)

5 Review of self-tuning regulator

- -Basic structure of self-tuning regulator
- -Minimum variance control
- -simulation of self-tuning control with simnon

(3 WEEKS)

6 Computer-aided Control System Design Applications

- -A generalized chemical processing model for digital control
- -Experience with fluidized bed reactor modeling and control
- -Experimental results in the modeling and control of a small furnace
- -Implementation of self-tuning controller for DC drive
- -Case studies in Aircraft Identification

(3 WEEKS)

COREQUISITES

Enrollment in "Stochastic Signals and Systems"
and "System Analysis Techniques"
or equivalent background.

System Identification and ~~Adaptive~~ self-tuning Control

The course will be directed to study the basic approaches of the model building and parameter estimation of dynamic systems using experimental data. Many results from probability theory and statistics will be used to develop the basic estimation algorithms. The published literature representing state-of-the-art of identification methods to practical problems will be reviewed.

The course description is summarized in the following:

1. Linear and Nonlinear Dynamic Models

   Fundamentals of the model building, classification of models based on linearity of parameters, input-output description, auto regressive moving average models, state-space cannonical models, Volterra series representation.

2. Least-squares and Normal Theory

   Development of least-squares algorithm. Properties of best linear unbiased estimator and minimum variance estimator. Numerical solution of normal equations using the Householder transformation.

3. Maximum Likelihood Estimation

   The likelihood function and the maximum likelihood estimator. Properties of the ML estimator.

4. Estimation of Linear Dynamic Models

   Development of the least-squares algorithm.
   Comparison with correlation methods.
   Problem of correlated residuals and biased estimates.
   Generalized least-squares.
   Instrumental variable method.
   Prediction error formulation and maximum likelihood estimation.

5. Parameter Estimation with Test Signals

   Impulse, step and sinusoidal test inputs.
   Pseudorandom binary input sequences.
   Time domain design of optimum input signals.

6. On-Line Parameter Estimation

Recursive least-squares.
Time-varying parameters and exponential window.
Application of Kalman filter method.
Approximate maximum likelihood.
Stochastic Approximation Algorithm.

7. Design of Self-Tuning Regulators

Basic structure of the self-tuning regulator.
Minimum variance control.
Convergence properties of the self-tuning algorithm.
Review of known applications.

8. Non-linear System Identification

Correlation-Regression method for bilinear system identification.
Estimation of Volterra kernels using pseudorandom test sequences.

9. Application of Identification Methods

Modeling and identification of a nuclear reactor.
Experimental results in the modeling and control of a small furnace.

Recommended Textbook

G. C. Goodwin and R. L. Payne, "Dynamic System Identification: Experimental Design and Data Analysis," Academic Press, 1977.

Library References

1. P. Eykhoff, "System Identification, Parameter and State Estimation", John Wiley and Sons, 1974.

2. R.K. Mehra and D.G. Lainiotis, "System Identification: Advances and Case Studies", Academic Press, 1976.

3. G. E. P. Box and G. M. Jenkins, "Time Series Analysis: Forecasting and Control," Holden-Day, 1976.

Appendix C

THE FEDERATED COMPUTER-AIDED CONTROL DESIGN SYSTEM

H. Austin Spang, III
GE Research and Development Center
Schenectady, NY 12345

The advent of the microprocessor has changed the economics of applying modern control theory to a variety of industrial processes providing an opportunity for significant performance improvements. The challenge is the selection and reduction to practice of the most appropriate control algorithm for a particular application. A cost effective approach to control system design requires computer-aided design tools. Without these tools, the cost of exploring alternatives, of answering the "what if" questions, and generating and testing the control software becomes prohibitive.

In this report, the structure and operation of the Federated Computer-Aided Control Design System will be discussed. The system is termed "Federated" to indicate that it consists of several independently developed subsystems tied together by a unified data base. In this manner, one takes advantage of existing software while providing the user with a unified system that spans the entire control design problem: modeling, design, simulation and implementation. While numerous Computer-Aided Control Design packages exist [1], most are focused on a particular aspect of the design problem. The Federated System is unique in the way it ties diverse packages together into a unified system.

1.0 Federated Approach

In creating a control system design, the engineer must construct models for the process to be controlled, analyze their behavior, design an appropriate control strategy and evaluate its overall performance. Eventually, he will implement the design in appropriate hardware such as a microprocessor. A brief non-inclusive summary of the design procedure and some of the current techniques which might be used is shown in Figure 1. The goal of the Federated System to provide an engineer with a broad spectrum of alternative design approaches.

The control design software shown in Table 1 are the initial major subsystems within the Federated structure. As indicated in Figure 1, these subsystems provide most of the desired design capability: IDPAC for modeling, CLADP and SSDP for analysis and design, and SIMNON for non-linear simulation. It is expected that subsystems for non-linear design and adaptive control will be added. In addition, the Federated system includes programs to go from one subsystem to another. These include linearization of the SIMNON non-linear model, and generation of parameter dependent non-linear control algorithms. As will be discussed in more detail in Section 3, the Federated system allows additional software to be added on-line. Thus, the user can customize the system to meet his specialized requirements with, for example, additional graphic display programs or specialized design algorithms.

The structure of the Federated System is designed to meet the following objectives:

1. Each subsystem can be operated as a stand-alone program.

2. Subsystems and other programs can be added to the system easily.

3. Subsystems or programs can be modified without affecting other parts of the system.

4. Federating adds a minimum amount of overhead to each subsystem.

5. User commands can be added easily and are valid for specified subsystems.

To meet these objectives, the Federated System is organized in a hierarchical structure of stand-alone subsystems connected by a supervisory program. A block diagram of the system is shown in Figure 2. The supervisor primarily serves as an operating system interface translating user commands into the names of programs that will be run. It also passes initialization and file information to allow that program to startup correctly. Once a subsystem is entered, control is not returned to the supervisor until the user enters a command that is handled by another subsystem. Thus the federating of the subsystems generates no overhead except during the transition from one subsystem to another.

It should be recognized that any subsystem may also be organized as a series of stand-alone programs connected by a supervisory program. This approach further enhances the modularity and maintainability of the system. Both the Cambridge Linear Analysis and Design Package and the State Space Design Package have been organized in this manner.

2.0 Data Base Management

One of the major problems in tying several subsystems together is to provide a common unified data base which describes the user's plant and his associated control system. The user must be able to enter a description of his plant in many ways, either from measurement data or linear or non-linear models. Once entered, the user must be able to go from one subsystem to another without having to reenter any of the previously entered or generated information.

Each subsystem,however, has its own way of handling the information it needs. Since each subsystem has been developed independently of the others, the way data is handled reflects the developer's insight into the problem and tradeoffs determined by his computer system. Generally, this data handling is integral to the subsystem. Any attempt to force a common data base structure would result, essentially, in a complete rewrite of that subsystem.

In the Federated System, this problem of data exchange is handled in one of two ways. The first takes advantage of the similarity of the information and is shown in Figure 3. This approach is useful in those situations where the information required is the same but the format is different. A common set of files forms the data base. Each subsystem reads that information through a set of subroutines forming a 'data base manager' for that subsystem. These subroutines read the information in the standard format and pass it on in the form required by that subsystem. Thus the change in the form of the data base is invisible to the subsystem. This approach is particularly useful for those subsystems requiring the same matrices.

The second approach to data exchange is by direct data conversion programs. The supervisor calls a conversion program before executing the next subsystem. A block diagram of this approach is shown in Figure 4. The conversion process maybe invisible to the user or may interact to determine which files should be converted. The advantage of this approach is that it can handle widely diverse forms of information. The interaction between SIMNON and the other subsystems is handled in this manner. For example, a conversion routine generates linear models from the non-linear models. Another takes the feedback designs from CLADP or SSDP and generates SIMNON code blocks for non-linear simulation.

A block diagram of the necessary data conversions between the initial subsystems of the Federated system is shown in Figure 5. The straight lines indicate that no conversion is necessary. The primary conversions are those between SIMNON and the design subsystems. The rest are basically format changes.

3.0 Supervisor

The supervisor primarily serves as an interface between the operating system and the user. For each user command, the supervisor will pass initiization information and run a sequence of programs. A more detailed block diagram of the interaction between the supervisor and a typical subsystem is shown in Figure 6. At the time a subsystem is installed in the Federated system, the user defines one or more "known global commands" which are used to execute that subsystem. The user also indicates other subsystems which should recognize or know about these commands. These command names are stored in the supervisor's "known global command" file as well as information on the sequence of conversion and subsystem programs to be run to execute that command. Due to its potential size, the information to initialize these programs is stored in a separate random access "PASS" file. The supervisor does not directly send the initialization information but rather sends pointers to the location of this information in the PASS file.

Each subsystem has its own "known global command" file containing the names of the global commands that will be recognized by that subsystem providing an extended set of commands beyond those normally recognized by the subsystem. When the user issues a command string, the known global commands are searched first for a possible match so the user has the capability of redefining any subsystem command by defining a corresponding global command. When a global command is found, control is passed back to the supervisor to initiate the sequence of programs associated with that command.

3.1 Supervisor Commands

To facilitate the installation of user commands and subsystems, the supervisor provides a set of commands which are described in this section.

INSTALL {<command name>} [<formal arguement>|
<delimiter>|<termination marker>]* {<subsystem>}*

The command name is inserted into the known global command file for the supervisor. The first subsystem name indicates the subsystem that handles the command. It must be given. If other subsystems are specified, the name is also inserted in that subsystem's known global command file. The supervisor inserts the subsystem name in its SYSTEM table and creates a known global command file if the subsystem has previously not been specified.

To maintain on-line self documentation, the supervisor returns a "HELP>" prompt message and waits for input immediately after the install command is entered. The user must supply at least two non-blank lines. The first is a one line summary of what the command does. It is typed whenever the HELP command is given. The second line is the start of a more detailed discussion which will be typed when the command HELP <command name> is given. Additional lines of text can be given. The sequence is terminated if a line is given containing only a [CR].

The install command is followed optionally by one or more CONV, PASS, or RUN commands. Previously defined user commands or file names may also be included. If a command is not a supervisor command or a previously defined user command, the command is assumed to be a file name for a macro containing additional commands. The install command sequence is ended by an END command.

In a manner similiar to INTRAC macros, formal arguments can be specified and used in subsequent command lines. When issuing the command, the user provides values which are substituted for the formal arguments wherever they occur.

The install command can also be used to add a previously defined command to a subsystem. In this case the install command consists of the single command line and is not followed by the end command. An attempt to install a previous command in a subsystem which already knows about that command results in an error message. Following an install command for a previously defined command with additional information also results in an error.

END

Indicates the end of an install sequence. It implies that the supervisor will return to the previous command sequence or prompts the user for more commands.

If the user issues the end command, control returns to the supervisor which will abort all previous command sequences and prompt the user for more commands.

CONV {<filename>} {ASK|DEF|<filename>} {<to subsystem>}
{ASK|DEF|<filename>}

Converts data files from one subsystem to another. The conversion program given by the first filename is run. The supervisor passes to that program the remaining arguements. These define what input/output files should be used:

- ASK - Conversion program should ask the user for the file name.
- DEF - Conversion program uses a default convention to define the file names.
- filename - conversion program uses this name for the input or output files. This name maybe one of the formal parameters included in the command definition.

PASS {<string>|<filename>}*

The PASS command provides a means of conveying information to a program. The given string or filename is passed to the program defined by the subsequent CONV or RUN command. Pass commands defined prior to the CONV command are sent to the conversion program. Those defined after the CONV command are sent to the program defined by the RUN command. If more than one PASS command is given, the strings are passed in the order defined. Strings can consist of any alpha-numeric ASCII characters except commas. Formal parameters are replaced by their values, provided the parameters are separated from the rest of the string by commas. Otherwise the supervisor does not interpret these strings. The format and meaning of the data depends on the subsystem.

RUN {<filename>}

Defines the program to be run in order to execute the command. If more than one run command is present, the programs are executed in the order given.

<filename> [parameters|delimiters]*

If the filename is not a supervisor or previously defined user command, the supervisor will read the next commands from the specified file. Full INTRAC macro capability is available.

REMOVE {<command name>} [subsystem]*

The command name and its associated data structure is removed form the supervisor known global command file. The name is also removed from all subsystem known global command files and the help file. If a subsystem name is given, the command name is removed from that subsystem's known global command file. In this case, the associated data structure and help file information is not removed until the command is removed from the supervisor known global command file.

LIST [ALL|FULL|<subsystem name>]*

List the commands in the supervisor known command file. If a subsystem name is given, it lists the names in that subsystem's known global command file. If the option ALL is given, all of the supervisor and subsystem commands are listed. The option FULL is used to list the associated data structure of each of the supervisor known commands.

HELP [<command name>]

Lists the help information associated with that command. If a command name is not given, all of the commands in the supervisor known global command file and a one line summary of each command is listed.

DEV {TTY|PRI|SHOW}

Changes the device used by the help and list commands:

TTY - Ouput goes to the users terminal. (default)
PRI - Output goes to the printer file.
SHOW - Prints at the user's terminal the current device.

DEFAULT {<varaible>=}* <argument>

The indicated assignment is made only if the named variable is "unassigned" through a formal parameter or previous assignment statement.

3.2 Supervisor Data Structure

The data structure used in the supervisor is shown in Figure 7. The known global command table holds the names of all the commands known to the Federated system. Corresponding to each entry in the known global command table are three other entries: a subsystem designation word, a pointer to the start of that command's Install Sequence Table entry, and the number of entries in the sequence table associated with this command. The subsystem designation word contains a bit for each subsystem which has this command in its own Known Global Command table. The name of the subsystem corresponding to each bit is kept in the SYSTEM table. This designation word is used by the REMOVE, HELP, and LIST commands.

The Install Sequence Table contains an integer entry corresponding to each command line making up the definition of the user command as given in the INSTALL command. The integer entry is used to branch to the supervisor code to perform that supervisor command line. For each entry, there is also an entry in the file table. The entry in the file table contains the name of the macro file to be read or the name of the file containing the executable code for the RUN and CONV commands.

Due to its potential size, the information contained in the PASS command string is kept in a random access file. The entry in the file table is a pointer to the information on this random access file. These pointers are sent to the subsystem executing the RUN or CONV command. The subsystem reads the random access file to obtain the passed information. If the entry in the file table is a file name, the name is passed to the subsystem indicating a separate pass file.

4.0 Subsystem Interface

The subsystem interface to the federated system consists of two subroutines: an initialization subroutine and a command search subroutine. These are linked through a common area containing that subsystem's known global commands. The initialization subroutine is called immediately after the subsystem is initiated. It first reads the supervisor mailbox. If there is no message, the subroutine assumes that the subsystem as been started stand-alone and sets a flag to avoid searching the known global commands. If a message is received, the subroutine reads in the known global commands for that subsystem and the PASS file information. It then initials the subsystem based on the passed information.

The command search subroutine searches the known global command table each time a user command is given. If the given command is found in the known command table, the subroutine returns control to the supervisor by sending the user command string through the supervisor mailbox and terminating execution of the subsystem. It is generally desirable that the global search be done before the local command interpretation. This allows the user to custom design command sequences some of which may have the same names as a local command.

5.0 Overview of Operation

In this section, a summary of how the Federated System operates is given. To use the Federated system, the user first runs the supervisor. The supervisor initializes its known global command table and associated data tables from it known global command file. A mailbox with a name known to all subsystems is also created. It then sends a prompt to the users terminal. The user is now able to issue any supervisor command or any previously defined user command. Assume that a user command is issued. The supervisor locates the command in its known global command table. Using the associated pointer and number of entries, the entry in the Install Sequence Table is located. For each entry in the Install Sequence Table, the supervisor branches to code to execute that command. A summary follows:

- PASS - Supervisor writes the pass file pointer to the mailbox.
- CONV - Supervisor writes the convert arguments to the mailbox and runs the program designated by the filename.
- RUN - Supervisor runs the program designated by the filename.

When the supervisor initiates a program, it goes to 'sleep' waiting for a message to be written to the mailbox.

When the subsystem is initiated, the initialization subroutine is called. This subroutine first attempts to read the supervisor mailbox. If there is no message for this subsystem, a flag to prevent global command searching is set and the subroutine returns to the main part of the subsystem. If a message exists, the subroutine reads the pass file using the pointers sent through the mailbox. This information is then used to initialize the subsystem. The format of the information in the pass file and its effect depend completely on the subsystem. The initialization subroutine also reads that subsystem's known global command file. When a user command is found in the known global command table, the command search routine writes the command string to the supervisor mailbox and exits the subsystem.

The writing of the message in the mailbox by the subsystem, causes the supervisor to be reactivated. If the user command string indicates a termination of the subsystem with the commands EXIT, STOP or FED, the supervisor continues the execution the current command sequence. If the command END is given, the supervisor aborts the current and all previous command sequences. If a user command is given, the supervisor executes that command before continuing with the previous sequence. At the end of any command sequence the supervisor returns to the previous sequence or prompts the user for more commands.

6.0 Conclusion

The Federated Computer-Aided Control Design System provides a means of providing a broad range of design techniques. The system is a loosely coupled set of programs with a unified data base. The basic modularity enhances the maintainablity and expandability of the system. A happy result is that the user can define on-line new commands customized to his application.

References

1. Frederick, D.K. (1981), "Computer Packages for the Simulation and Design of Control Systems", Arab School on Science and Technology, 4th Summer Session, September, 1981, Bloudan, Syria

2. Edmunds,J.M. (1979), "Cambridge Linear Analysis and Design Programs", IFAC Symposium on Computer Aided Design of Control Systems, Zurich, 253-258.

3. Wieslander,J. (1979), "IDPAC User's Guide, Revision 1.", Report 7605, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

4. Elmquist, H. (1975) "SIMNON, an Interactive Simulation Program for Nonlinear Systems" Report 7502, Department of Automatic Control, Lund, Sweden.

5. H. A. Spang,III, (1981) "State Space Design Program, (SSDP)" GE Internal Report., December, 1981.

TABLE I

COMPUTER-AIDED CONTROL SYSTEM DESIGN PROGRAMS

USED IN THE FEDERATED SYSTEM

SOURCE	NAME	AUTHOR	FUNCTION OF PROGRAM
Cambridge University, England [2]	Cambridge Linear Analysis and Design Program (CLADP)	Prof. A.G.J. MacFarlane	Multivariable Control System design by frequency domain methods
Lund University, Sweden [3]	IDPAC	Prof. K.J. Astrom	System identification
Lund University, Sweden [4]	SIMNON	Prof. K.J. Astrom	Nonlinear simulation
General Electric [5]	State Space Design Package (SSDP)	H.A. Spang	Multivariable control system design by state space and time domain methods

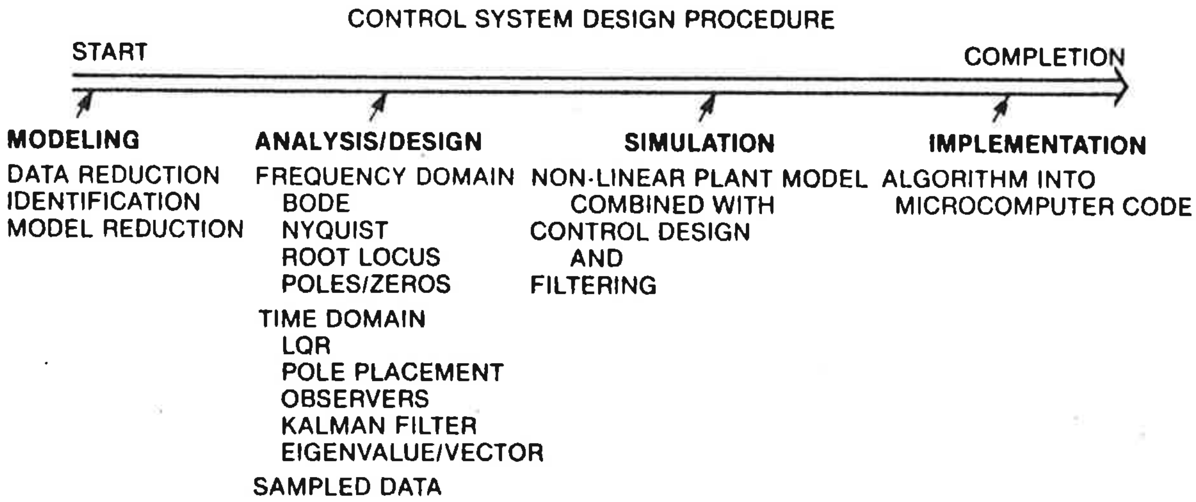
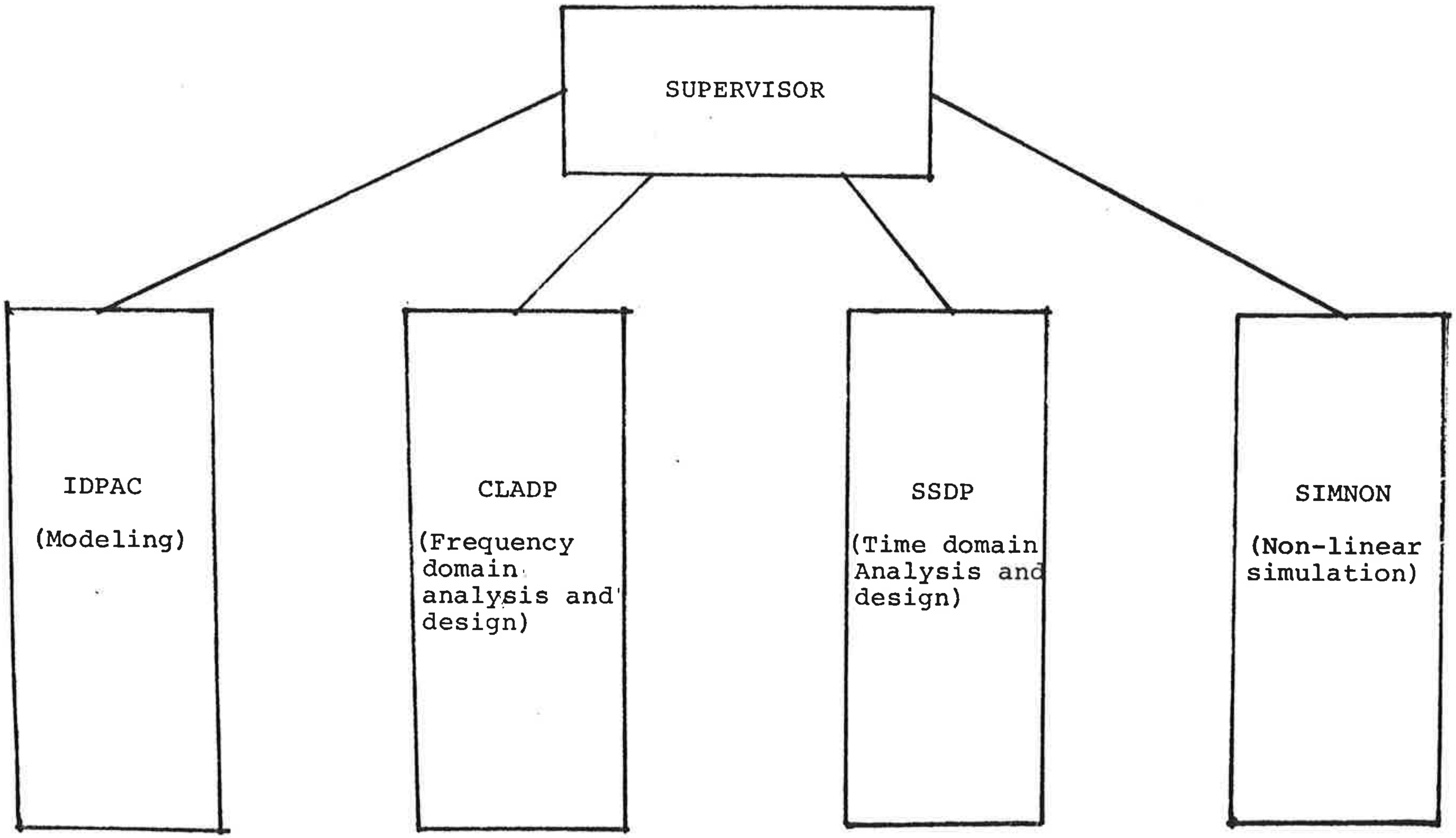


Figure 1.
Summary of Techniques Used in the Different Stages of a Control Design

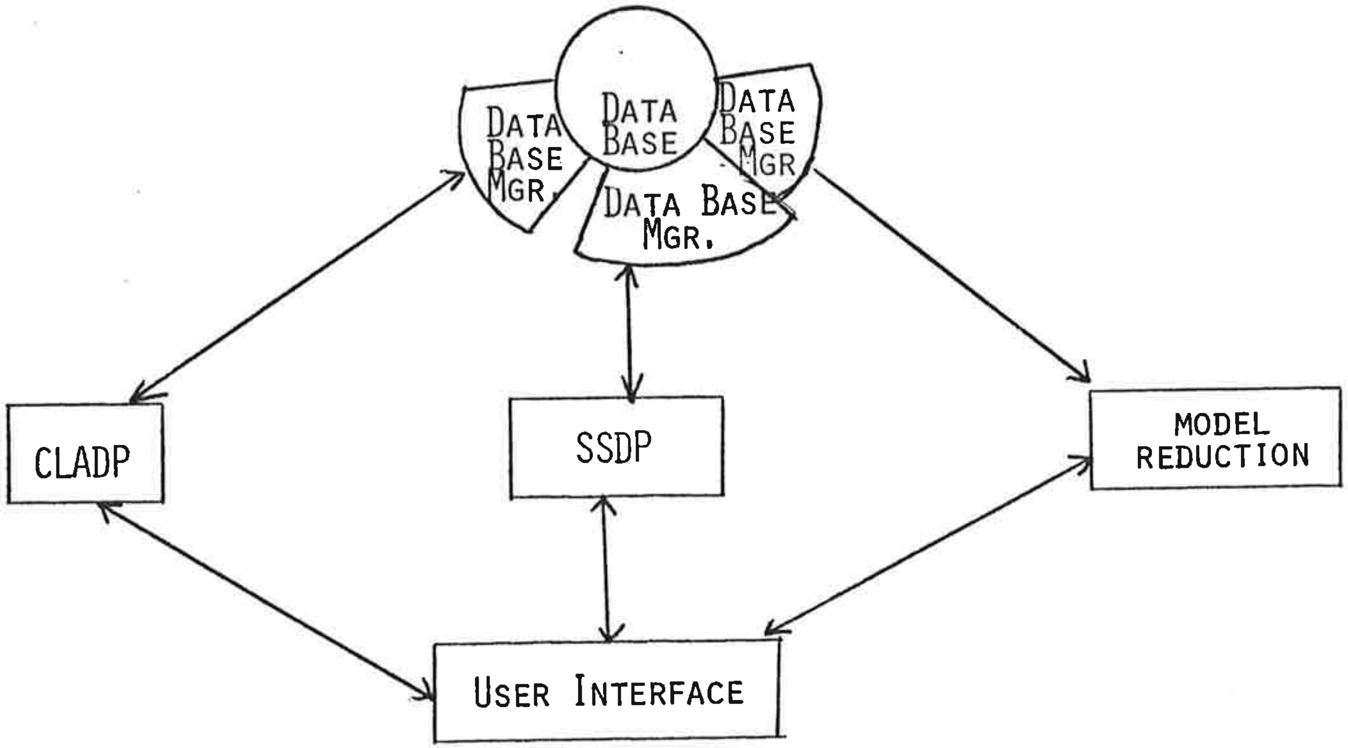


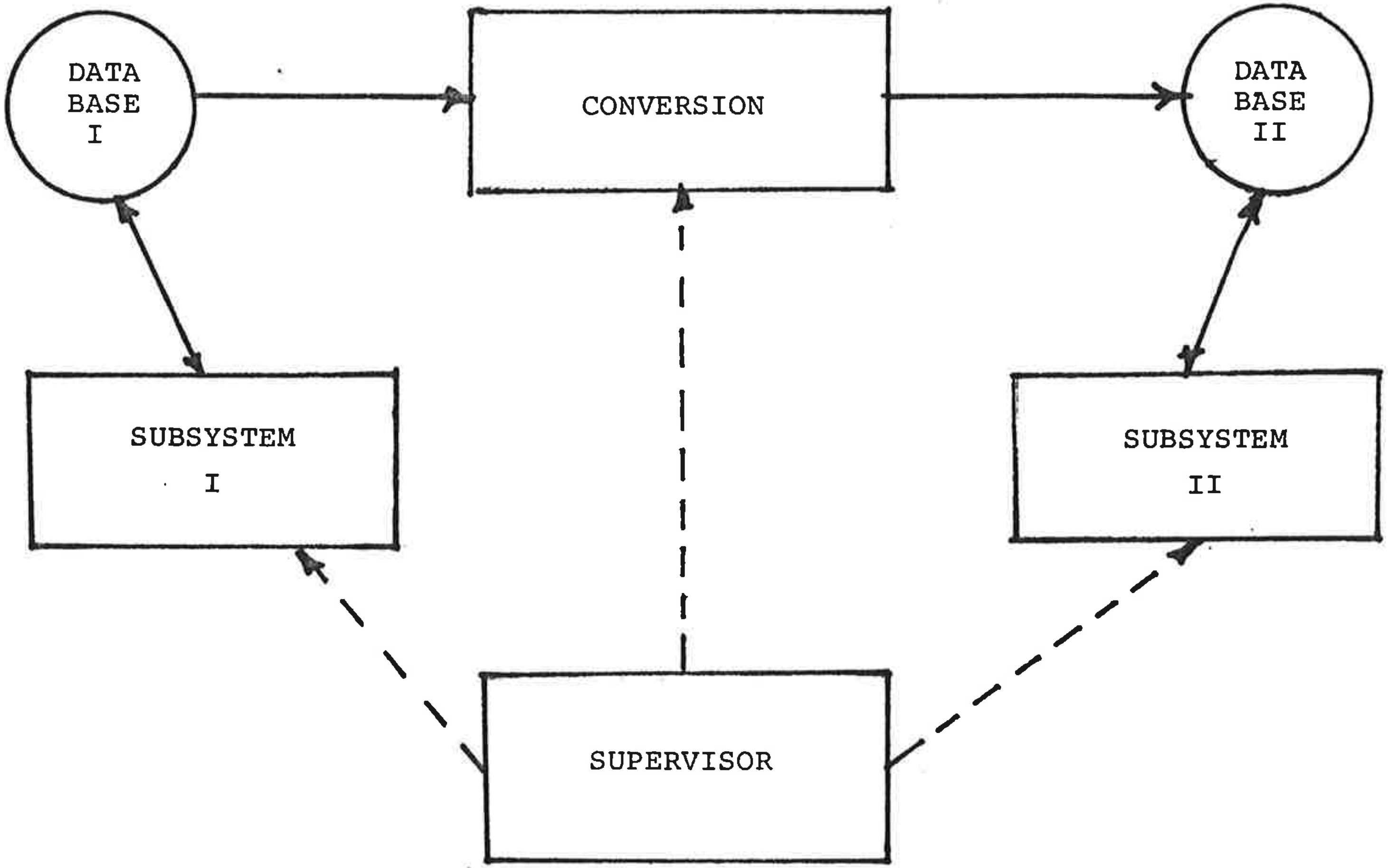
THE FEDERATED SYSTEM

FIGURE 2

DATA FILE HANDLING

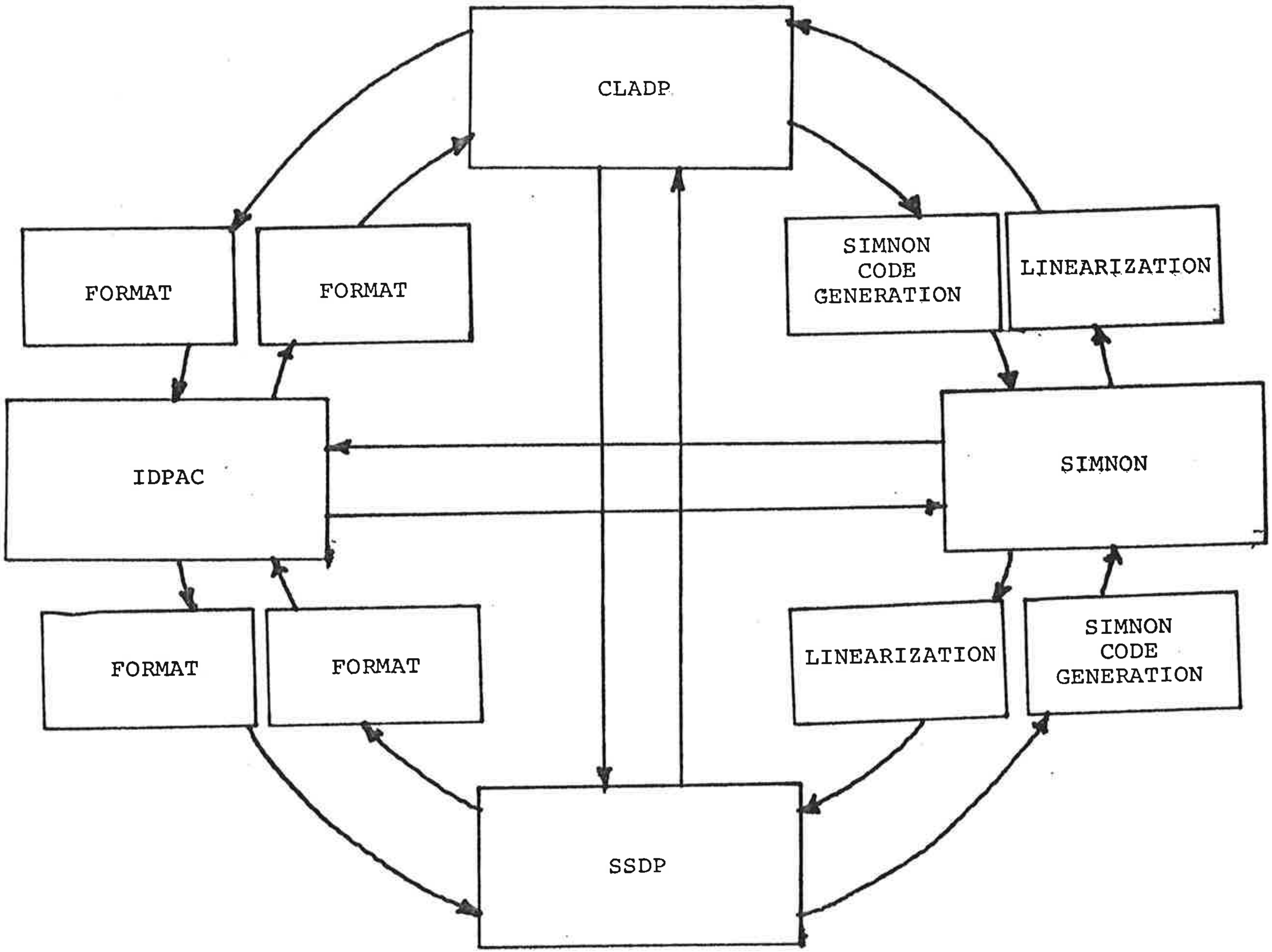
COMMON DATA BASE APPROACH





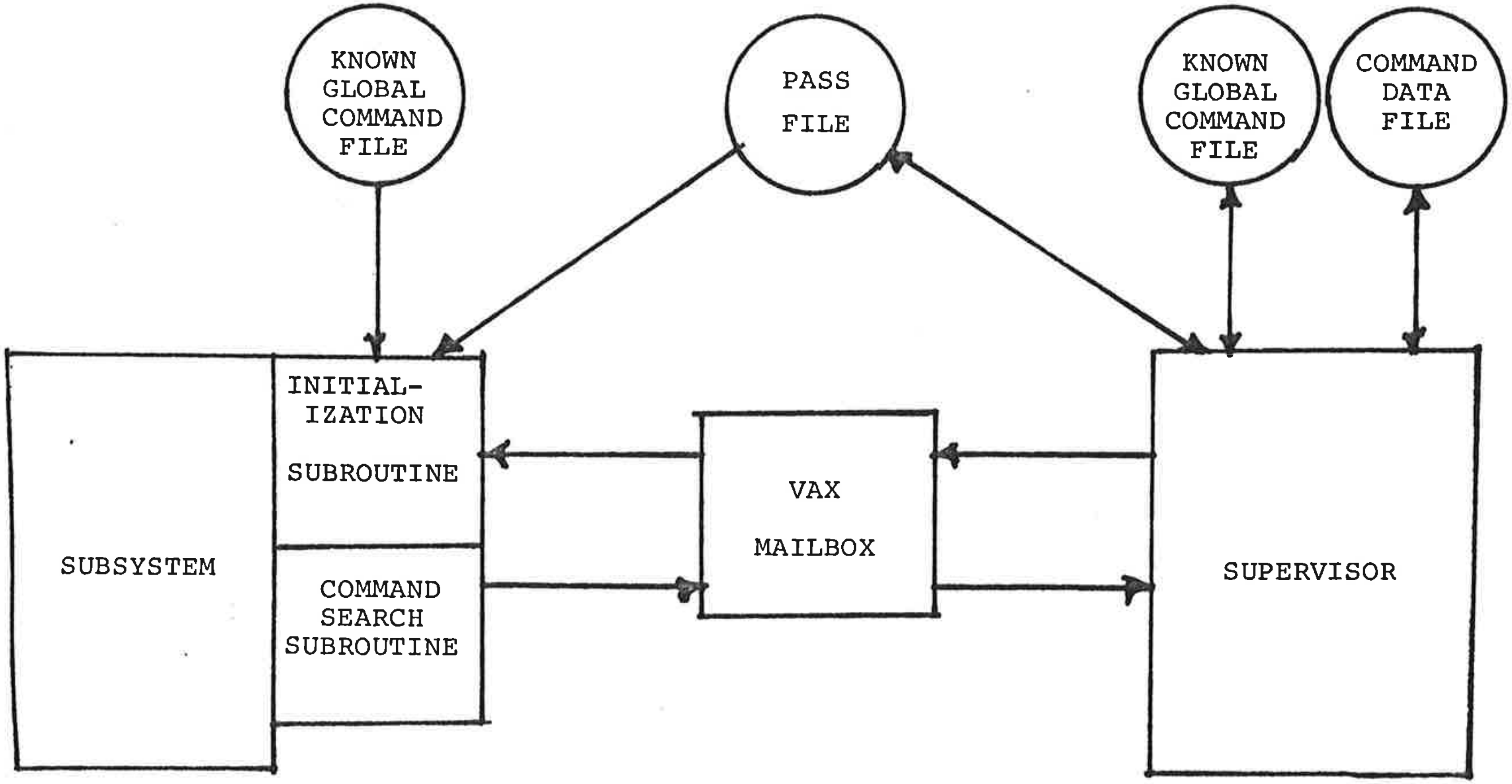
DATA CONVERSION

FIGURE 4



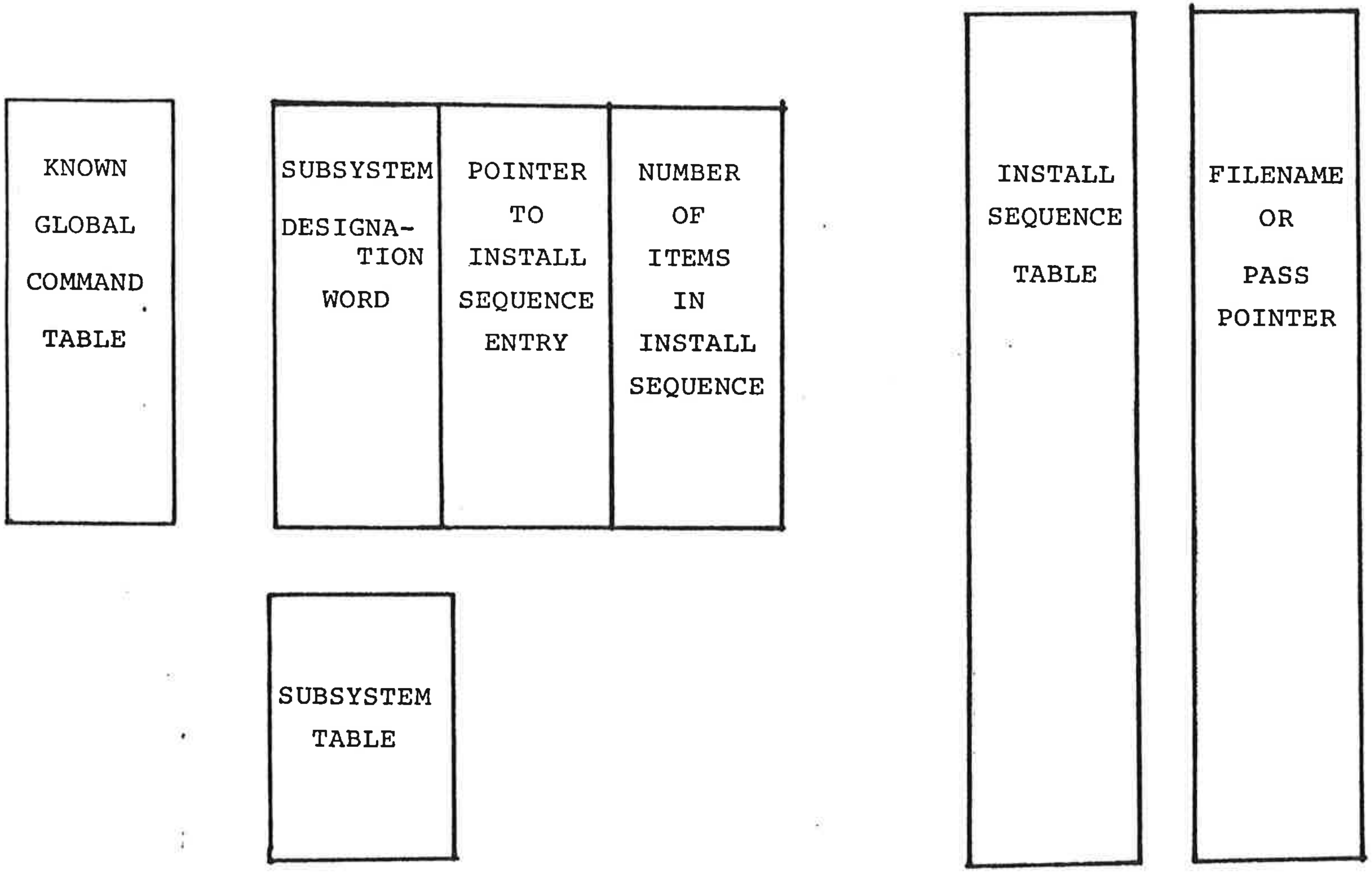
DATA CONVERSIONS BETWEEN SUBSYSTEMS

FIGURE 5



SUPERVISOR AND SUBSYSTEM INTERACTION

FIGURE 6



SUPERVISOR DATA STRUCTURE

FIGURE 7

Appendix D

SIMNON COMMAND SYNTAX   D. KASSOVER 3 JAN 1983

Below is a table containing the syntax for the new commands recently added to the SIMNON program.

SYSIN [<VARIABLE>... [-ADD]]
SYSOUT [<VARIABLE>... [-ADD]]
SYSSTATE [<VARIABLE>... [-ADD]]

SYSIN	specifies input variables to be considered when finding equilibrium or linearizing
SYSOUT	specifies output variables to be considered when finding equilibrium or linearizing
SYSSTATE	specifies state variables to be considered when finding equilibrium or linearizing

If a system name is specified (e.g. [NREAC]) then all of the appropriate variables associated with that system are included. If -ADD is specified, the specified variable(s) are appended to the list of variables specified by previous commands, if any, otherwise, the list is reset before adding the specified variable(s) to the list. If no variable or system is specified, then the list is reset, with no entries.

If no SYSSTATE command is specified, then the LINEAR command, when invoked, will use ALL of the states in ALL of the systems.

Examples:

SYSIN U1[SYS1]	the input list is reset, and the variable U1 associated with system SYS1 is entered on it.
SYSIN [SYS1]	the input list is reset, and all the input variables associated with system SYS1 are entered on it.
SYSIN U3[SYS2] -ADD	the variable U3 associated with system SYS2 is entered on the input list

EQUIL [T]

Requests that an equilibrium point be calculated. The inputs specified by the previous sysin command are disconnected from their driving functions. Equilibrium is calculated at time T, or at time 0. if T is not specified

Examples:

EQUIL	Calculate an equilibrium point at time = 0.
EQUIL 1.234	Calculate an equilibrium point at time = 1.234.

DEL <VARIABLE>:(<NUMBER>|<VARIABLE>)

Sets the delta to be used by the linearization process for the variable specified. If a variable is specified as [SYSTEM], then all the variables associated with that system will have their deltas set to the value specified. During linearization, if the delta has not been set explicitly, a default is assumed.

Examples:

DEL X1[SYS1]:.34	the delta associated with variable X1 in system SYS1 is set to .34
DEL [SYS2]:.02	the deltas for all variables associated with system SYS2 are set to .02
DEL X5[SYS3]:U1[SYS1]	the delta for variable X5 in system SYS3 is set to the current value of the variable U1 in system SYS1

LINEAR <FILENAME> [-(AUTO,MAN)]

Requests that the current system as defined by the SYSIN, SYSOUT, and SYSSTATE commands be linearized. FILENAME is the name of a file into which the results are placed. If it is not specified, it is prompted for. If -AUTO is specified, or no option is specified, the deltas are adjusted to minimize truncation or roundoff error in the linearization calculation. If -MAN is specified, the deltas are used exactly as found in the table. SYSIN and SYSOUT are required to have been invoked previously. If SYSSTATE has not been invoked, then the linearization will involve ALL of the states in ALL of the systems.

DISP [<DEVICE SPEC>] (<VARIABLE>...:ABCD:EQUI:LERR)

displays, on the device chosen by <DEVICE SPEC>, either the specified variables, the current linearization results, the current equilibrium conditions, or the current linearization errors. If <VARIABLE> is specified as [system], then all variables associated with that system are displayed. <DEVICE SPEC> is one of the currently supported SIMNON devices.

Examples:

DISP X1[SYS1]	displays variable X1 associated with system SYS1
DISP [SYS1]	displays all variables associated with system SYS1
DISP ABCD	displays the current contents of the A, B, C, D matrices (linearized system)
DISP EQUI	displays the current equilibrium conditions
DISP LERR	displays the current linearization errors (that is, the difference between the slope of the function at +/- delta and +/- 2*delta

Appendix E

ESTIMATION AND FILTERING PANEL CHARTER

- FILTERING AND PREDICTION
  - REAL TIME DATA PROCESSING
- SMOOTHING
  - POST EXPERIMENT/FLIGHT DATA PROCESSING
- IDENTIFICATION
  - STATE AND PARAMETER

APPLICATIONS

- TRACKING
- NAVIGATION
- ATTITUDE DETERMINATION
- SENSOR CALIBRATION
- PARAMETER ESTIMATION AND IDENTIFICATION
- PRE EXPERIMENT/FLIGHT PLANNING
- POST EXPERIMENT/FLIGHT ANALYSIS
- FAULT DETECTION
- MAINTENANCE ASSESSMENT
- ADAPTIVE BEAMFORMING
- IMAGE ENHANCEMENT

INTERACTIVE KALMAN FILTER ANALYSIS AND DESIGN PACKAGE

OBJECTIVE (PHASE I) :

DEVELOP BASIC CAPABILITIES TO ANALYSIS, DESIGN AND SIMULATE EXTENDED-KALMAN FILTERS

APPROACH :

- USE INTERACTIVE CAPABILITY OF SIMNON
- USE BIERMAN ESTIMATION SUBROUTINE LIBRARY

1 MODEL DEFINITION AND INITIALIZATION

- CONTINUOUS TIME NONLINEAR SYSTEM
- DISCRETE NONLINEAR MEASUREMENTS

$$\frac{d}{dt}\underline{x}(t) = \underline{f}(x(t), t) + G(t)\,\underline{w}(t)$$

$$\underline{y}(t_k) = \underline{h}(x(t_k), t_k) + \underline{v}(t_k) \qquad t \geq t_0$$

INITIAL CONDITIONS $\underline{x}(t_0)$, $P(t_0|t_0)$

NOISE STATISTICS $\varphi(t)$, $R(t)$

- USE SIMNON LANGUAGE TO DEFINE SYSTEM MODEL

LINEAR TIME-INVARIANT DYNAMIC MODEL

$$\frac{d}{dt}\,\underline{x}(t) = A\,\underline{x}(t) + B\,\underline{u}(t) + D\,\underline{w}(t)$$

$$\underline{y}(t_k) = C\,\underline{x}(t_k) + \underline{E} + \underline{v}(t_k)$$

SEPARATE SET OF INPUT MODULES

- READ MEAUREMENTS FROM DATA FILE

BASIC CAPABILITIES : (PHASE I)

1 MODEL DEFINITION AND INITIALIZATION

2 LINEARIZATION

3 CONTINUOUS TO DISCRETE TRANSFORMATION

4 MEASUREMENT UPDATE ALGORITHM

5 TIME UPDATE ALGORITHM

6 OPTIMAL UPDATE AND COVARIANCE COMPUTATION

7 FREQUENCY RESPONSE OF THE KALMAN FILTER

USE SIMNON CAPABILITIES FOR

- NON UNIFORM SAMPLING OF MEASUREMENTS
- DIFFERENT START TIMES AND DURATION FOR EACH SENSOR
- SCALING OF MEASUREMENTS AND OUTPUTS
- RESET OF FILTER DURING RUN TIME
- PRINTING AND PLOTTING OF KEY STATES AND MATRICES

2. LINEARIZATION

- ON LINE LINEARIZATION OF NONLINEAR MODEL ABOUT CURRENT STATE ESTIMATE

- USER DEFINED PARTIAL DERIVATIVES (JACOBIAN MATRICES)
  - ANALYTICAL FORMULA
  - NUMERICAL VALUES

3. CONTINUOUS TO DISCRETE TRANSFORMATION

- USE SOFTWARE FROM SSDP FOR LINEAR TIME-INVARIANT MODEL

- NEW SOFTWARE FOR LINEAR TIME-VARYING DYNAMIC SYSTEM AND COVARIANCE MATRICES

4. MEASUREMENT AND TIME UPDATE ALGORITHM

- U D FACTORIZATION

- USE BIERMAN'S ESTIMATION SUBROUTINE LIBRARY

6. OPTIMAL ESTIMATE AND COVARIANCE COMPUTATION

(USE SPECIAL STRUCTURE OF KALMAN EQUATIONS)

- U D FACTORS TO COMPUTE OPTIMAL GAINS
- SHUR'S METHOD TO COMPUTE STEADY STATE GAINS
- USER SUPPLIED GAINS FOR SUB OPTIMAL FILTER DESIGN

7. FREQUENCY RESPONSE OF KALMAN FILTER

- INTERFACE WITH CLADP TO GENERATE ABCD MODEL OF KALMAN FILTER AT GIVEN TIME
- DETERMNE FILTER BANDWIDTH

FEA Estimation Subroutine Library

Directory Summary*

U–D Kalman Filter Related Subroutines

CNS/CNSX	colored noise smoothing subroutines
COV2UD/UD2COV	covariance/U–D conversions
D2U/U2D	diagonal/U array conversions
INVTST	K–σ innovations test
MAPU	mapping of U array with covariance output
PINVB/PINVEK	smoother gain related equation solvers, P*Y = V
RANK1/RNK1	rank–1 U–D modification
SFU/STPHIU/PHIU/TPHIU	PHI*U subroutines
SGDCMP	U–D matrix σ decomposition
UCOL	colored noise time update of U array
UCON/UDOPT/UMEAS	U–D fator measurement updates
U2SIG	computes σ's from U–D factors
WGSG	Generalized Gram–Schmidt time propagation

SRIF Least Squares Related Subroutines

COV2RI/RI2COV	covariance/SRIF array conversions
INF2R	Information matrix to SRIF array conversion
RA	R*A, SRIF deterministic map
RCOLRD/TRCOL	SRIF colored noise time updating
RINCON/UTINV/UTIROW	upper triangular matrix inversions
RINZ	equation solver R*Y = Z
RL2RU/RU2RL	copy an upper triangular subarray
RTMSX	evaluate R*X
SINTST	SRIF K–σ innovations test
THHC/TDHHT/TTHH	Householder matrix triangularization

Miscellaneous (General Purpose) Subroutines

C2C	reorder rows and columns of covariance matrix
COVCOR	compute correlation matrix from the covariance
HHPOST	Householder covariance square–root triangularization
MNPRNT/TRIMAT	Matrix printing routines
PERMUT	reorder columns of a rectangular matrix
R2A	reorder columns of a triangular matrix
RUDR	R to U–D and U–D to R conversion
TZERO	Triangular matrix row zeroing
WXY2U	copies WXY matrix into corresponding WXY locations.

---

* as of 3/01/82

Appendix F

EXPERT SYSTEM ARCHITECTURE FOR COMPUTER-AIDED CONTROL SYSTEM DESIGN

Dr. James H. Taylor
Control Technology Branch

Prof. Dean K. Frederick
Department of Electrical, Computer, and Systems Engineering
Rensselaer Polytechnic Institute

Electrical, Computer, & Systems Department Seminar

23 February 1984

CACSD-III-1

OVERVIEW

* INTRODUCTION AND MOTIVATION

* EXPERT SYSTEMS CONCEPTS FOR THIRD-GENERATION CACSD

* EXPERT SYSTEMS OVERVIEW (DKF)

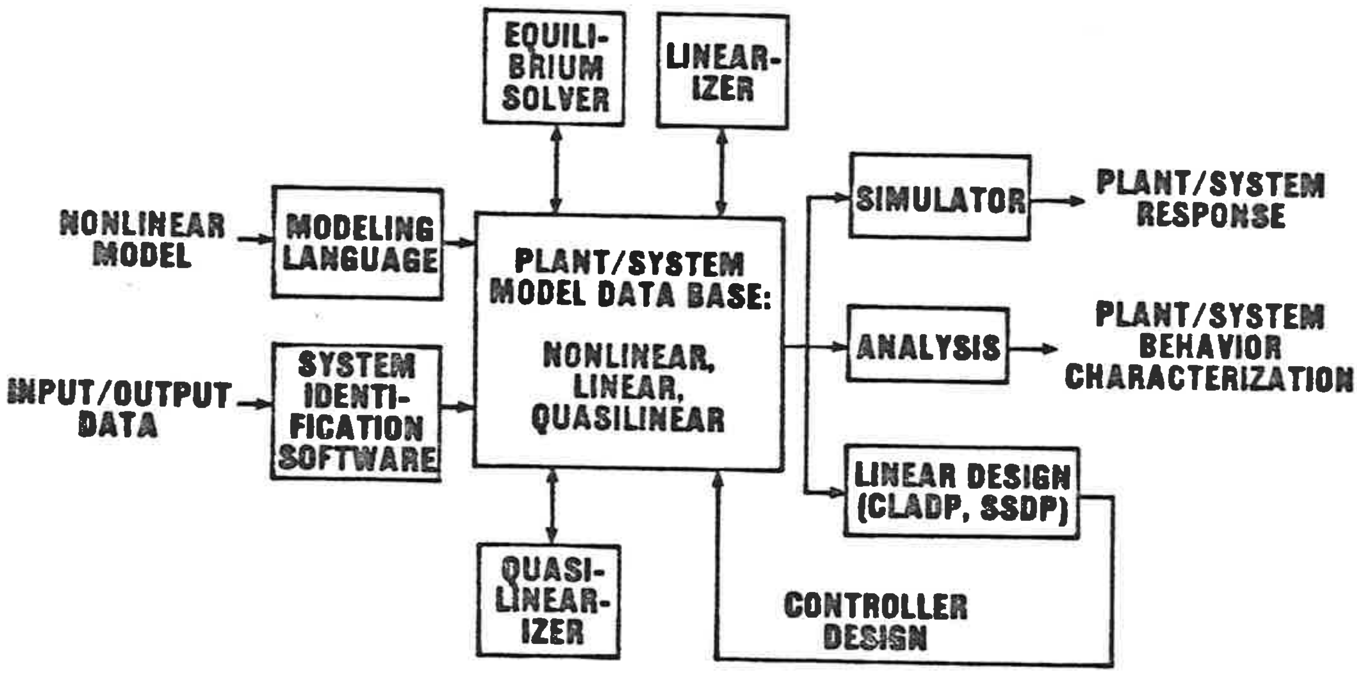
* SESSION ILLUSTRATION (DKF)

* STATUS, SUMMARY, AND CONCLUSIONS

CACSD-III-2

INTRODUCTION & MOTIVATION

* CACSD IS A VERY BROAD PROBLEM:



... ADD RIGOROUS VALIDATION AND IMPLEMENTATION AND ...

CACSD-III-3

INTRODUCTION & MOTIVATION (CONT'D)

* SECOND-GENERATION CACSD SOFTWARE IS POWERFUL BUT ...

  > LOW LEVEL

  > UNSUPPORTIVE

  > DIFFICULT TO USE

* THIS SITUATION WILL GET WORSE

CACSD-III-4

MOTIVATION & INTRODUCTION (CONT'D)

* PRESENT PROBLEMS:

  > PACKAGES WITH INCOMPLETE SCOPE

  > LOW-LEVEL ENVIRONMENT:

  SIMU Ø 1ØØ .1 /RUN1

  ROOT; CA; Ø.; Ø.

  > LITTLE OR NO GUIDANCE AND SUPPORT

  IS THE PROBLEM WELL-POSED?

  WHAT DESIGN METHOD(S) ARE BEST?

  WHAT NEXT?

  > LITTLE OR NO USEFUL DOCUMENTATION

  DESIGN JUSTIFICATION

  VALIDATION

* THE FUTURE:

  > NEW APPROACHES ADDING TO THE BURDEN

  - LINEAR
  - NONLINEAR

  > MORE OF THE 'TOTAL PROBLEM' BEING CONSIDERED IN CACSD - E.G., IMPLEMENTATION

CACSD-III-5

CACSD-III - WHAT CAN BE DONE?

PROVIDE A HIGHER-LEVEL, MORE SUPPORTIVE ENVIRONMENT WITH BUILT-IN EXPERTISE FOR:

> MODEL DEVELOPMENT

> PROBLEM FORMULATION

> DESIGN APPROACH SELECTION

> TRADEOFF ANALYSIS

> VALIDATION

> IMPLEMENTATION

> KEEPING TRACK OF THINGS

> USING CACSD SOFTWARE EFFECTIVELY

CACSD-III-6

A CENTRAL CONCEPT - THE PROBLEM FORMULATION

(A 'LIST OF FACTS' OR 'FRAME')

MODEL CHARACTERISTICS

* HOW NONLINEAR
* POLE LOCATIONS
* ZERO LOCATIONS
* RESONANCES
* CONTROLLABILITY & OBSERVABILITY
* ETC.

SPECIFICATIONS

* RISE TIME
* BANDWIDTH
* PERCENT OVERSHOOT
* ETC.
* SENSITIVITY
* ROBUSTNESS

CONSTRAINTS

* A / D IMPLEMENTATION
* A: ORDER
* D: SAMPLING TIME
* DECENTRALIZATION
* ETC.

CACSD-III-7

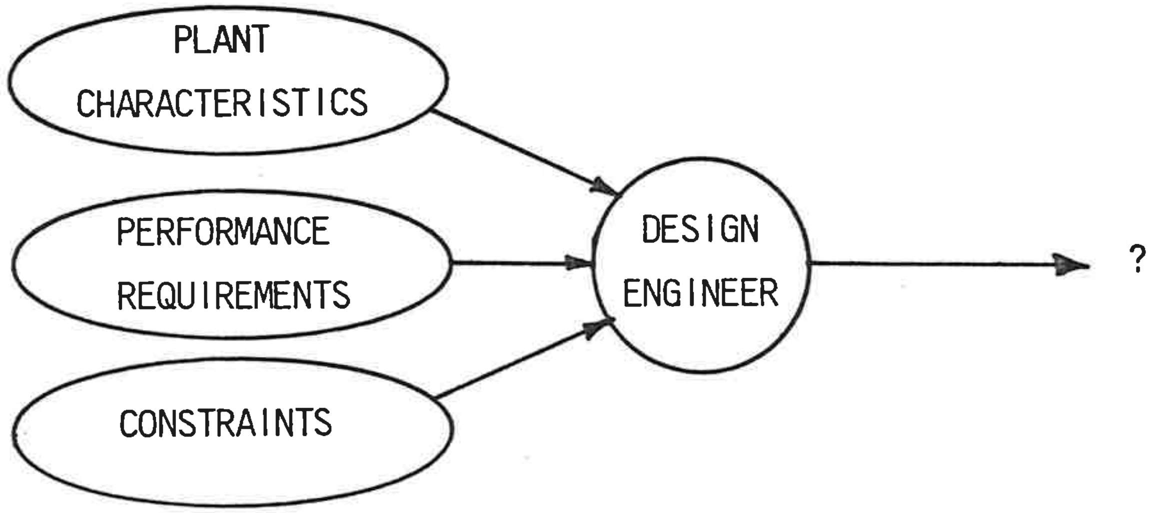
EXPERTISE

THE CONTROLS EXPERT . . .

* FORMULATES THE FRAME
* DEBUGS THE FRAME
  > COMPLETENESS
  > CONSISTENCY
  > WORKABILITY
* ASSESSES ITS DIFFICULTY
* DETERMINES NEEDS (BASED ON FRAME)
* MATCHES THESE NEEDS WITH AVAILABLE DESIGN PROCEDURES (DPs)
* EXECUTES APPROPRIATE DPs
* PERFORMS TRADE-OFFS IF NECESSARY
* VALIDATES THE DESIGN
* DOCUMENTS THE DESIGN
* IMPLEMENTS THE DESIGN

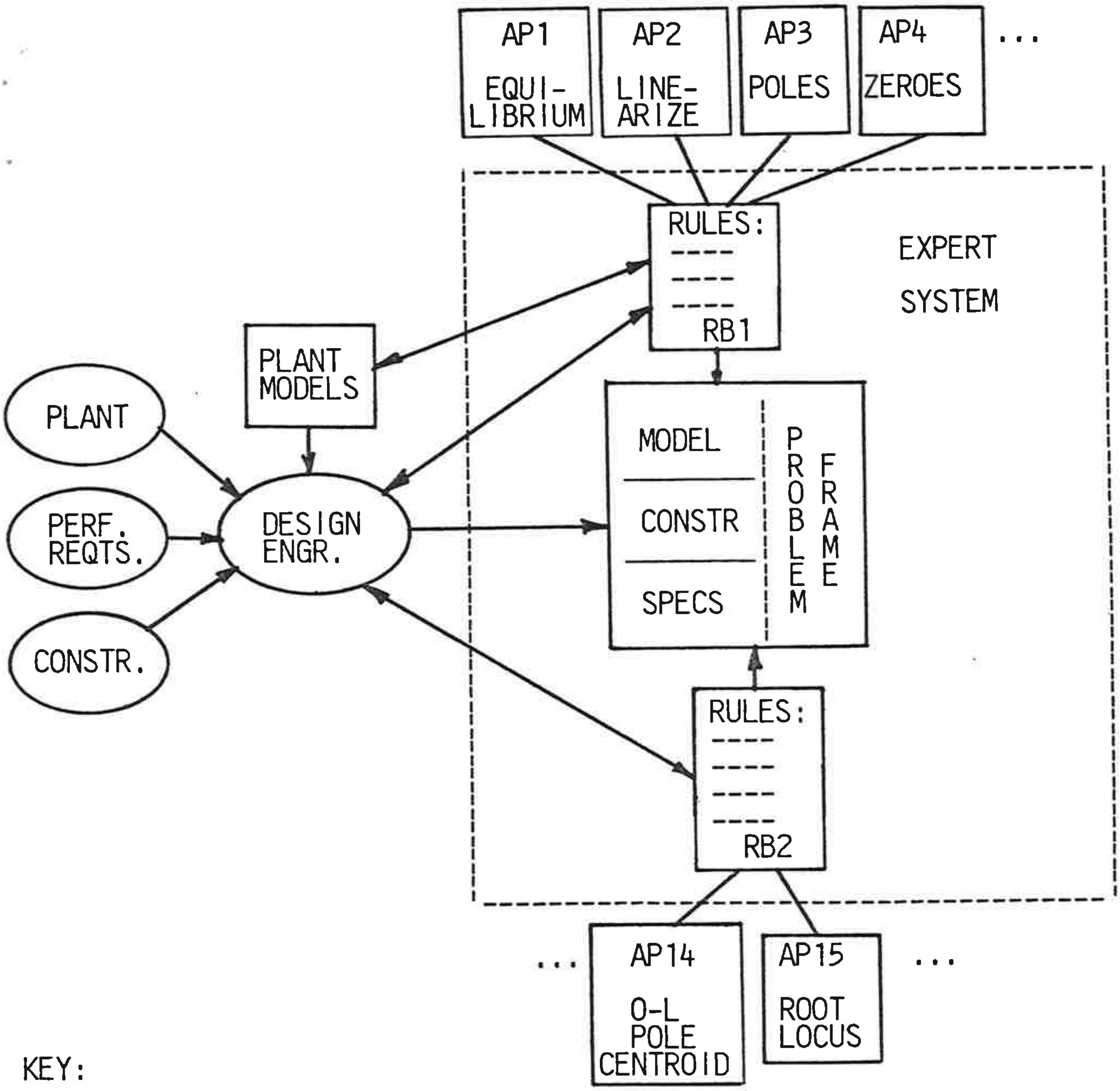
CACSD-III-8

THE INITIAL CONDITION



CACSD-III-9

EXPERT SYSTEMS SUPPORT FOR PROBLEM FORMULATION

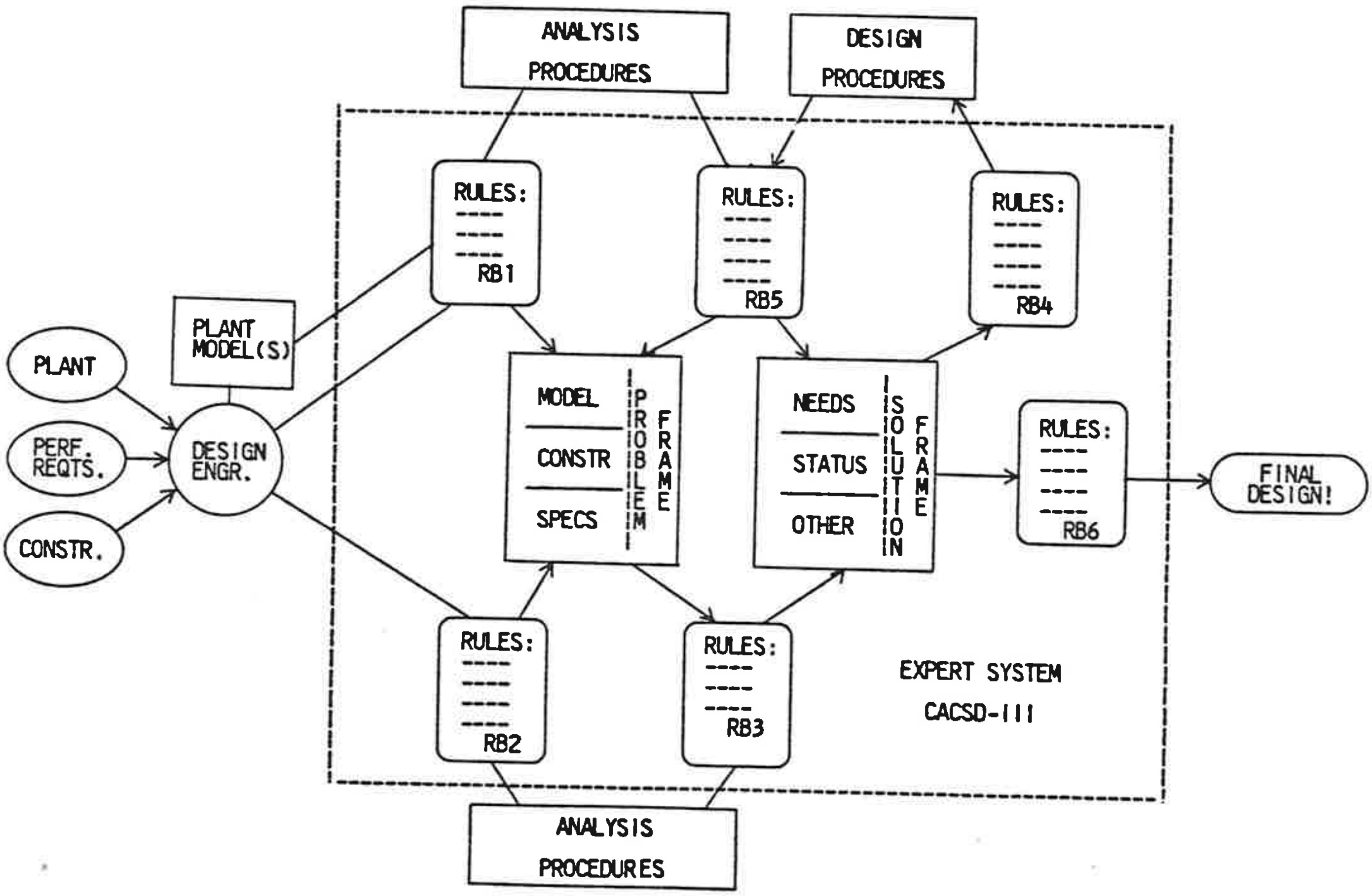


KEY:

APN = ANALYSIS PROCEDURE N
RBN = RULE BASE N

CACSD-III-10

CACSD-III: COMPLETE FUNCTIONAL STRUCTURE



CACSD-III-11

CACSD-III: RULE BASE FUNCTIONS

RB1 - PROVIDES SUPPORT IN MODEL DEVELOPMENT; WRITES ALL REQUIRED PLANT DATA INTO THE KNOWLEDGE BASE

RB2 - GUIDES THE USER IN DEVELOPING SPECIFICATIONS (CHECKS FOR CONSISTENCY, COMPLETENESS, AND REALISM)

RB3 - LOOKS AT THE SPECIFICATIONS, CONSTRAINTS, AND PLANT CHARACTERISTICS AND DECIDES WHAT NEEDS TO BE DONE

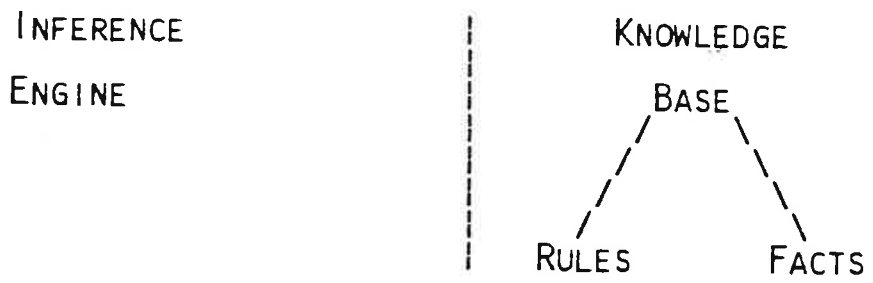
RB4 - SELECTS DESIGN PROCEDURES

RB5 - UPDATES THE SOLUTION FRAME TO REFLECT THE EXECUTION OF DESIGN PROCEDURES; SUPERVISES TRADEOFF ANALYSIS

RB6 - GOVERNS FINAL CONTROL SYSTEM VALIDATION AND REDUCTION TO PRACTICAL IMPLEMENTATION

CACSD-III-12

EXPERT SYSTEMS



FACTS

[OBJECT ATTRIBUTE VALUE]

EXAMPLES:

[GAIN-MARGIN VALUE ADJUSTED]

[LEAD QUANTITY ONE]

[BANDWIDTH VALUE 4<...<5-RPS]

[DESIGN-FACTS VALUES ALL-ENTERED]

RULES

PREMISE (SITUATION RECOGNITION)

- CONDITION ON STATE OF KNOWLEDGE BASE

CONCLUSION (ACTION)

- CHANGES TO BE MADE IN KNOWLEDGE BASE

```
RULE Ø1Ø6 ( WANT TO ENTER MAX RE PART SPEC; ALREADY ASSIGNED )

IF:
   EQ [ SPEC-C-L-POLE  MAX-REAL-PART  REQUESTED ]
      ( USER HAS ASKED TO ENTER MAX-REAL-PART )
   EQ [ MAX-REAL-PART  VALUE  +  ]
      ( A VALUE IS CURRENTLY ASSIGNED )
THEN:
   CLR [ SPEC-C-L-POLE  MAX-REAL-PART  REQUESTED ]
      ( RESET - CLEAR THE FACT THAT TRIGGERED THIS RULE )
   DISPLAY [ + + + ]
      ( YOU WANT TO ENTER A VALUE FOR THE MAXIMUM )
      ( REAL PART OF THE POLES, BUT A VALUE HAS BEEN )
      ( ASSIGNED PREVIOUSLY. )
   ASK [ MAX-REAL-PART  MODIFY  VALUE ]
      ( DO YOU WISH TO REPLACE THE CURRENT VALUE? [Y OR N] )

RULE Ø1Ø8 ( CURRENT VALUE OF MAX RE PART IS TO BE REPLACED )

IF:
   EQ [ MAX-REAL-PART  MODIFY  VALUE ]
      ( USER WANTS TO MODIFY THE CURRENT VALUE )
THEN:
   CLR [ MAX-REAL-PART  MODIFY  VALUE ]
      ( RESET - CLEAR THE FACT THAT TRIGGERED THIS RULE )
   CLR [ MAX-REAL-PART  VALUE  + ]
      ( DELETE THE OLD VALUE )
   WRITE [ SPEC-CL-POLE  MAX-REAL-PART  UNASSIGNED ]
      ( RESET - PAVE THE WAY FOR A NEW ASSIGNMENT )
   WRITE [ SPEC-CL-POLE  MAX-REAL-PART  REQUESTED ]
      ( TRIGGER THE RULE TO REQUEST ENTRY OF NEW VALUE )
```

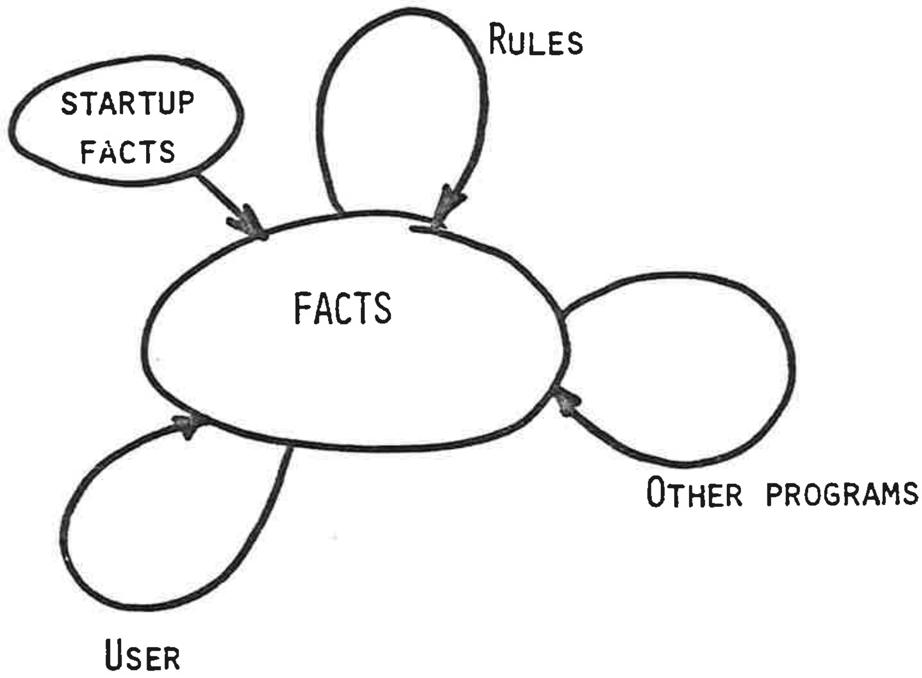
EXPERT SYSTEMS - WHAT ARE THEY? (CONT'D)

LIST OF FACTS AFTER A DIAGNOSE AND SPECIFY SESSION

FACT	BASIS
PLANT MODEL NONLINEAR	USER-ASKED
PLANT-NL-MODEL FNAME EXOREACT	USER-ASKED
PLANT-NL-MODEL TIME-TYPE CONTINUOUS	INFERRED
PLANT-NL-MODEL STATE-TYPE CONTINUOUS	INFERRED
PLANT-NL-MODEL ORDER 2	INFERRED
PLANT-NL-MODEL INPUTS 2	INFERRED
PLANT-NL-MODEL OUTPUTS 2	INFERRED
PLANT-NL-MODEL DIAGN-DATA-FNAME EXORNDATA	INFERRED
PLANT-NL-MODEL NL-BEHAVIOR MILD	INFERRED
PLANT-L-MODEL FNAME EXOREACTL	INFERRED
PLANT-L-MODEL STABLE NO	INFERRED
PLANT-L-MODEL CONTROLLABLE YES	INFERRED
PLANT-L-MODEL OBSERVABLE YES	INFERRED
PLANT-L-MODEL MINIMUM-PHASE YES	INFERRED
MODEL DIAGNOSIS DONE	INFERRED
SENSOR TIME-TYPE CONTINUOUS	USER-ASKED
CONTROLLER TIME-TYPE CONTINUOUS	USER-ASKED
CONTROLLER STRUCTURE DIAG-DOMINANT	USER-ASKED
CONTROLLER CHANNEL1-IN U1	USER-ASKED
CONTROLLER CHANNEL1-OUT Y1	USER-ASKED
CONTROLLER CHANNEL2-IN U2	INFERRED
CONTROLLER CHANNEL2-OUT Y2	INFERRED
MAX-STEP-SS-ERR CH1-VALUE Ø.ØØ5	USER-ASKED
MAX-REAL-PART CH1-VALUE -1.4	USER-ASKED
MIN-DAMPING-RATIO CH1-VALUE 1.Ø	USER-ASKED
MAX-STEP-SS-ERR CH2-VALUE Ø.ØØ5	USER-ASKED
MAX-REAL-PART CH2-VALUE -1.4	USER-ASKED
MIN-DAMPING-RATIO CH2-VALUE 1.Ø	USER-ASKED
CONTINUOUS-SPEC ENTRY DONE	USER-ASKED
CONTINUOUS-SPEC ENTRY REALISTIC	INFERRED
CONTINUOUS-SPEC ENTRY COMPLETE	INFERRED
CONTINUOUS-SPEC ENTRY CONSISTENT	INFERRED
SPEC-SESSION TERMINATION NORMAL	INFERRED
END OF FACTS.	

CACSD-III-17

CHANGES IN THE FACTS



CACSD-III - A SAMPLE TRANSACTION

```
WELCOME TO CACSD-III
RESETTING: FACTS, RULES
QUESTION: IS THE MODEL LINEAR OR NONLINEAR [L OR NL]       ... ENTER: NL
CONTROLLER DESIGN FOR A NONLINEAR PLANT MODEL
ENTER FILENAME OF PLANT MODEL FILE                          ... ENTER:EXOREACT
ENTER SIMULATE OR DIAGNOSE [S OR D]                         ... ENTER: D
ENTER DESIRED OPERATING VALUE OF U, UØ                      ... ENTER:50.,33.8
ENTER APPROXIMATE EQUILIBRIUM, XØ                           ... ENTER: 15., 55.
ENTER APPROX INPUT RANGE, + DU (DU = U - UØ)                ... ENTER: 3., 9.
```

CACSD-III-19

CACSD-III - A SAMPLE TRANSACTION (CONT'D)

DIAGNOSIS OF NONLINEAR PLANT EXOREACT:

I. QUANTITATIVE RESULTS:

A. EQUILIBRIUM CORRESPONDING TO SPECIFIED UØ:

XØ = 9.98, 49.Ø1

B. FOURIER ANALYSIS RESULTS FOR INPUT DU * SIN (2.T):

MAXIMUM PERCENT SECOND HARMONIC: Ø.Ø

MAXIMUM PERCENT THIRD HARMONIC: 3.2

C. LINEARIZED MODEL EIGENVALUES AT UØ, XØ:

Ø.56 + 1.23J
Ø.56 - 1.23J

D. LINEARIZED MODEL DISTORTION ANALYSIS RESULTS FOR STEP INPUTS OF AMPLITUDE DU:

MAXIMUM PERCENT RMS DISTORTION: 1.36

CACSD-III-2Ø

CACSD-III - A SAMPLE TRANSACTION (CONT'D)

II. QUALITATIVE RESULTS:

A. THE EXOREACT SYSTEM MODEL HAS NO DISCONTINUOUS NONLINEARITIES OVER THE INPUT RANGE SPECIFIED.

B. THE EXOREACT SYSTEM MODEL IS ONLY MILDLY NONLINEAR OVER THE INPUT RANGE SPECIFIED.

C. THE LINEARIZED MODEL IS QUITE ACCURATE OVER THE INPUT RANGE SPECIFIED.

D. THE EXOREACT SYSTEM LINEARIZED MODEL IS UNSTABLE.

E. THE EXOREACT SYSTEM LINEARIZED MODEL IS MINIMUM-PHASE.

F. THE EXOREACT SYSTEM LINEARIZED MODEL IS CONTROLLABLE AND OBSERVABLE.

END OF DIAGNOSIS.

CACSD-III-21

CACSD-III - A SAMPLE TRANSACTION (CONT'D)

THE LINEARIZED MODEL HAS BEEN WRITTEN TO FILE EXOREACTL.DAT

QUESTION: DO YOU WISH FURTHER DIAGNOSTICS [Y OR N] ... ENTER: WHY

HELP FACILITY: ENTER ITEM NUMBER E.G. I.A ... ENTER: II C

BOTH THE NONLINEAR MODEL AND THE LINEARIZED MODEL HAVE BEEN SIMULATED WITH STEP INPUTS OF AMPLITUDE DU. THE LINEARIZED MODEL IS JUDGED TO BE ADEQUATELY ACCURATE AS A BASIS FOR CONTROLLER DESIGN IF THE MAXIMUM PERCENT RMS DISTORTION (LINEAR VERSUS NONLINEAR STEP RESPONSE) IS LESS THAN 5 %. RULE Ø414.

QUESTION: DO YOU WISH FURTHER DIAGNOSTICS [Y OR N] ... ENTER: N

ENTER CONSTRAIN OR SPECIFY [C OR S] ... ENTER: S

WARNING: WITHOUT CONSTRAINTS THE DESIGN YOU REALIZE MAY BE IDEALISTIC.

MENUS TO ENTER SPECIFICATIONS

MENU FOR SENSOR TYPE

1. CONTINUOUS-TIME.
2. DISCRETE-TIME.
3. MIXED
4. RETURN TO PRECEDING MENU. ... SELECT: 1

CACSD-III-22

CACSD-III - A SAMPLE TRANSACTION (CONT'D)

MENU FOR CONTROLLER TYPE:

1. CONTINUOUS-TIME.
2. DISCRETE-TIME.
3. MIXED
4. TERMINATE SESSION. ... SELECT: 1

MENU FOR CONTROL SYSTEM STRUCTURE

1. DECOUPLED (DIAGONAL DOMINANT).
2. DECENTRALIZED CONTROLLER
3. HIERARCHICAL
4. UNCONSTRAINED
5. RETURN TO PRECEDING MENU. ... SELECT: 1

MENU FOR DEFINING I/O PAIRS

1. DIRECT USER SELECT.
2. USER SELECT AFTER SEEING NYQUIST ARRAY.
3. CACSD-III ADVICE (BASED ON DIAGONAL DOMINANCE).
4. CACSD-III DIAGONAL DOMINANCE ALGORITHM.
5. RETURN TO PRECEDING MENU. ... SELECT: 1

ENTER FIRST I/O PAIR ... ENTER: U1, Y1

SECOND I/O PAIR [U2 Y2] INFERRED.

CACSD-III-23

SPECIFICATION DEVELOPMENT FOR FIRST I/O CHANNEL

MENU FOR TYPE OF CONTINUOUS-TIME SPEC

1. LOCATIONS OF POLES IN S-PLANE.
2. TIME RESPONSE.
3. FREQUENCY RESPONSE.
4. NO MORE SPECS TO ENTER.
5. RETURN TO PRECEDING MENU. ... SELECT: 1

MENU FOR TYPE OF S-PLANE SPEC

1. MAXIMUM REAL PART OF S.
2. MINIMUM DAMPING RATIO.
3. VALUE OF DOMINANT POLE [RE, IM].
4. DAMPING RATIO & NATURAL FREQ. OF DOMINANT POLES.
5. RETURN TO PRECEDING MENU. ... SELECT: 1

ENTER MAXIMUM REAL PART OF CLOSED LOOP POLES ... ENTER: -5.0

WARNING: YOUR SPEC FOR MAX REAL-PART OF CLOSED-LOOP POLES IS NOT REASONABLE. IT SHOULD NOT BE LESS THAN - 1.4

QUESTION: DO YOU WISH TO MODIFY? [Y OR N] ... ENTER: WHY

A POLE-PLACEMENT ALGORITHM HAS BEEN APPLIED TO THE LINEARIZED MODEL OF THE PLANT, AND IT HAS BEEN UNABLE TO PLACE THE POLES TO THE LEFT OF -1.4 WITHOUT EXCEEDING THE PLANT INPUT OPERATING RANGE YOU SPECIFIED.

CACSD-III-24

CACSD-III - A SAMPLE TRANSACTION (CONT'D)

QUESTION: DO YOU WISH TO MODIFY? [Y OR N] ... ENTER: Y

ENTER MAXIMUM REAL PART OF CLOSED LOOP POLES ... ENTER: -1.4

THIS SPECIFICATION IS REALISTIC.

MENU FOR TYPE OF CONTINUOUS-TIME SPEC

1. LOCATIONS OF POLES IN S-PLANE.
2. TIME RESPONSE.
3. FREQUENCY RESPONSE.
4. STRUCTURAL, E.G., NO. OF LEADS.
5. NO MORE SPECS TO ENTER.
6. RETURN TO PRECEDING MENU. ... SELECT: 1

MENU FOR TYPE OF S-PLANE SPEC

1. MAXIMUM REAL PART OF S.
2. MINIMUM DAMPING RATIO.
3. VALUE OF DOMINANT POLE [RE, IM].
4. DAMPING RATIO & NATURAL FREQ. OF DOMINANT POLES.
5. RETURN TO PRECEDING MENU. ... SELECT: 2

ENTER MIN DAMPING RATIO OF C.L. POLES ... ENTER: 1.0

THIS SPECIFICATION IS REALISTIC.

MENU FOR TYPE OF CONTINUOUS-TIME SPEC

1. LOCATIONS OF POLES IN S-PLANE.
2. TIME RESPONSE.
3. FREQUENCY RESPONSE.
4. STRUCTURAL, E.G., NO. OF LEADS.
5. NO MORE SPECS TO ENTER.
6. RETURN TO PRECEDING MENU. ... SELECT: 2

MENU FOR TYPE OF TIME RESPONSE SPEC

1. RISE TIME
2. PERCENT OVERSHOOT
3. SETTLING TIME
4. STEADY-STATE ERROR
5. RETURN TO PRECEDING MENU. ... SELECT: 4

YOUR SYSTEM IS TYPE ZERO.

ENTER PERCENT STEADY-STATE POSITION ERROR ... ENTER: 0.25

YOU HAVE NOW ENTERED A SET OF SPECIFICATIONS FOR CHANNEL 1 THAT APPEARS TO BE COMPLETE, CONSISTENT, AND REALISTIC.

CACSD-III-25

QUESTION: DO YOU WISH TO MODIFY OR ADD SPECS? [Y OR N] ... ENTER: N
YOU MAY ENTER SEPARATE SPECS FOR THE SECOND I/O CHANNEL, OR
USE THE SAME SPECS AS ENTERED FOR CHANNEL 1.

QUESTION: DO YOU WANT NEW SPECS FOR CHANNEL 2? [Y OR N] ... ENTER: N

YOU HAVE NOW ENTERED A SET OF SPECIFICATIONS FOR CHANNEL 2
THAT APPEARS TO BE COMPLETE, CONSISTENT, AND REALISTIC.

ENTER CONSTRAIN OR DESIGN [C OR D] ... ENTER: D

WARNING: WITHOUT CONSTRAINTS THE DESIGN YOU REALIZE
MAY BE IDEALISTIC.

YOUR SPECIFICATIONS SUGGEST USING THE HIGH-FREQUENCY
ALIGN (HFA) ALGORITHM TO ENSURE DIAGONAL DOMINANCE AT HIGH
FREQUENCIES. RECOMMENDED FREQUENCY IS 5.60.

QUESTION: DO YOU WISH TO USE HFA? [Y OR N] ... ENTER: Y

ENTER HFA FREQUENCY ... ENTER: 5.0

HFA COMPENSATOR DESIGN COMPLETED.

QUESTION: DO YOU WISH TO SEE HFA COMPENSATOR? [Y OR N] ... ENTER: Y

HFA @ 5.0 RPS - PRECOMPENSATOR GAIN MATRIX:

525.9101	-0.5813171
159.8909	100.7904

SPECIFICATIONS HAVE NOT BEEN MET COMPLETELY. HFA ACHIEVED:

HFA @ 5.0 RPS - CL POLES:

-4.926024
-5.030019

HFA @ 5.0 RPS - STEADY-STATE ERROR:

-2.810562 %
4.137923 %

CACSD-III-26

CACSD-III - A SAMPLE TRANSACTION (CONT'D)

REMAINING REQUIREMENTS SUGGEST USING THE APPROXIMATELY COMMUTATIVE CONTROLLER (ACC) DESIGN ALGORITHM TO DESIGN A LAG COMP TO REDUCE STEADY-STATE ERROR. BASED ON THE CHARACTERISTIC LOCI, THE RECOMMENDED ACC FREQUENCY IS Ø.85.

QUESTION: DO YOU WISH TO USE ACC? [Y OR N] ... ENTER: Y

ENTER ACC FREQUENCY ... ENTER: 1.0

IN ORDER TO MEET THE STEADY-STATE ERROR SPEC, IT IS RECOMMENDED THAT A LAG COMPENSATOR BE DESIGNED, WITH LOW FREQUENCY GAIN 1Ø.Ø AND CENTER FREQUENCY Ø.3Ø.

ENTER LAG COMPENSATOR CENTER FREQUENCY ... ENTER: 0.3

APPROXIMATELY COMMUTATIVE CONTROLLER DESIGN COMPLETED.

SPECIFICATIONS HAVE BEEN MET COMPLETELY. HFA + ACC ACHIEVED:

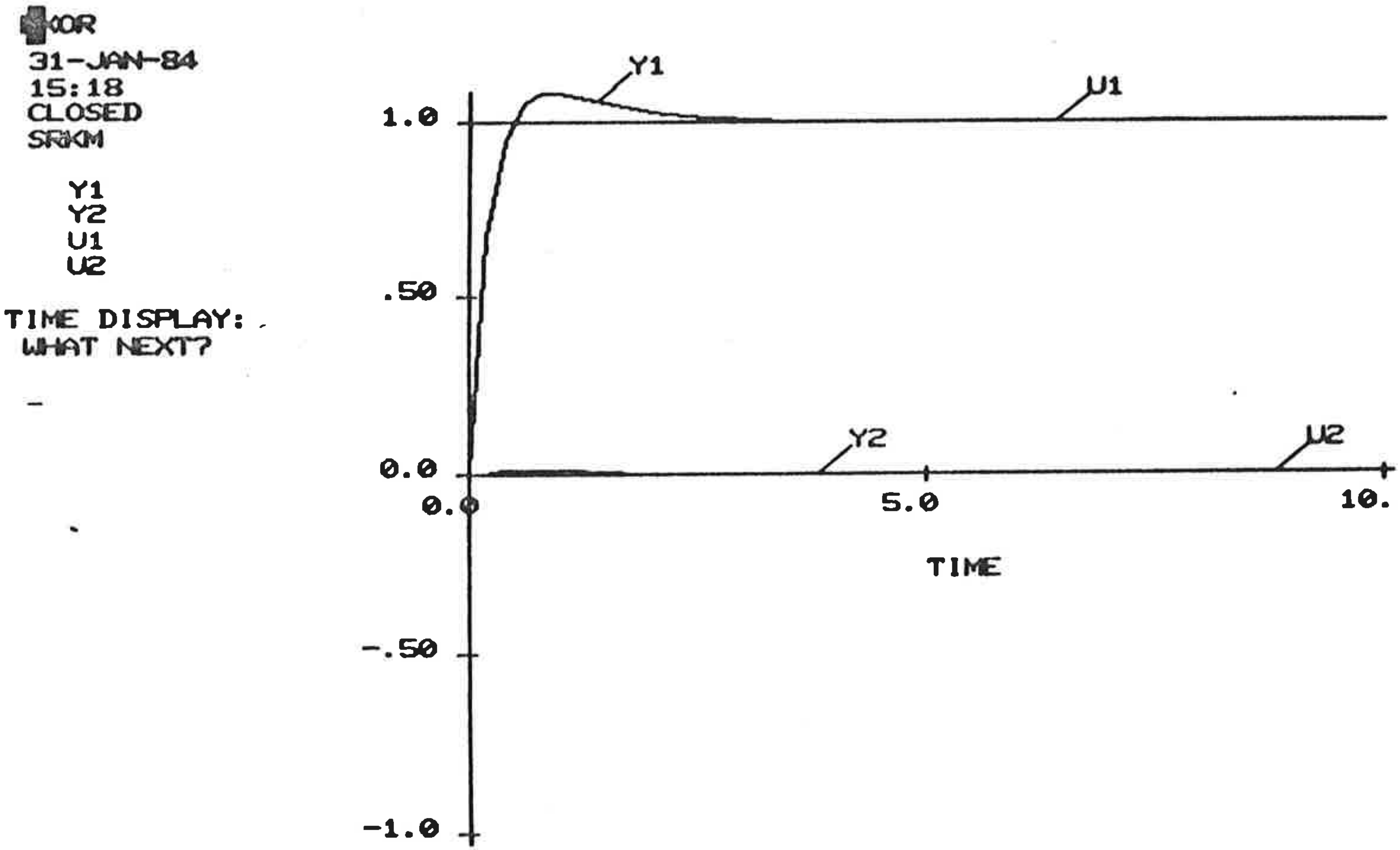
HFA @ 5.Ø RPS, ACC @ 1. RPS - CL POLES:

```
-1.3Ø175Ø
-1.389284
-3.634895
-3.83Ø161
```

HFA @ 5.Ø RPS, ACC @ 1. RPS - STEADY-STATE ERROR:

```
-Ø.281Ø56   %
 Ø.413792   %
```

QUESTION: DO YOU WISH TO SEE STEP RESP PLOTS? [Y OR N] ... ENTER: Y



EXOR HFA + ACC w/ $K(s) = \text{diag}\left[\frac{10s+10}{10s+1}\right]$

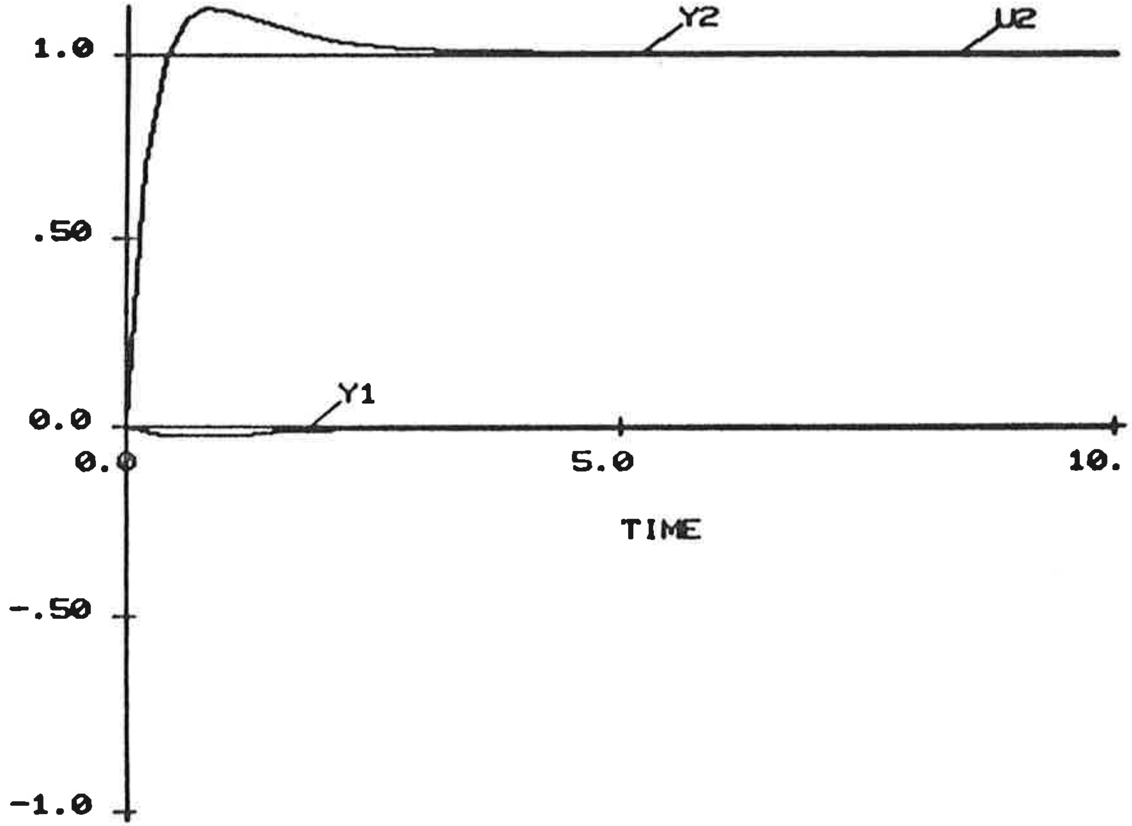
Step on U1

KOR
31-JAN-84
15:19
CLOSED
SRKM

Y1
Y2
U2

TIME DISPLAY:
WHAT NEXT?

—



Exor HFA + ACC w/ $K(s) = \text{diag}\left[\frac{10s+10}{10s+1}\right]$

CACSD-III - OTHER CONSIDERATIONS

* LISP OR FORTH ENVIRONMENT
  > RUNNING EXTERNAL PROCESSES
  > NUMERICS

* STEERABILITY
  > SUGGESTIONS - NO FORCE
  > SUPPORTED BY 'WHY' FACILITY
  > DIRECT FACT-WRITING:

```
MAX-STEP-SS-ERR  CH1-VALUE  Ø.ØØ5
MAX-REAL-PART  CH1-VALUE  -1.4
MIN-DAMPING-RATIO  CH1-VALUE  1.Ø
```

  > DIRECT ENTRY INTO COMMAND MODE (CLADP, ETC.)

CACSD-III-3Ø

CACSD-III - PROJECT STATUS

* FUNCTIONS AND FUNCTIONAL REQUIREMENTS DEFINED IN DETAIL

* SOFTWARE FOR NONLINEAR SYSTEM DIAGNOSIS DEVELOPED (SOME ROUGH)

* SOFTWARE FOR LINEAR SYSTEM EIGENVALUE DIAGNOSIS DEVELOPED

* RB FOR SPECIFICATION ENTRY UNDERWAY; IDEAS FOR DIAGNOSING SPECS PARTIALLY FORMULATED.

* ALGORITHM AND RB FOR LEAD/LAG COMPENSATOR DONE

CACSD-III-31

CACSD-III - SUMMARY

* ORIGINS IN DISCUSSIONS WITH A. G. J. MAC FARLANE

* STILL IN CONCEPT DEVELOPMENT

* 'VISION' IS QUITE TANGIBLE

* AT THE FOREFRONT OF THE TECHNOLOGY

* VERY FORTUNATE:
    > DELTA INFERENCE ENGINE
    > PIERO BONISSONE'S HELP
    > LT COL JOHN JAMES' WORK
    > NEAL LASSINGER, RALPH QUAN

--> ABLE TO GO WELL BEYOND 'MOCK-UP'

CACSD-III-32

CACSD-III - CONCLUSIONS

* EXPERT / KNOWLEDGE-BASED SYSTEMS CONCEPTS SHOW GREAT POTENTIAL FOR PROVIDING THE BASIS FOR A VASTLY IMPROVED MAN/MACHINE INTERFACE FOR CACSD.

* NO EXPERT SYSTEM WILL BE ABLE TO SOLVE "ALL PROBLEMS" -- OUR GOAL IS TO AUTOMATE CONTROLLER DESIGN WHERE POSSIBLE, AND PROVIDE ALL THE SUPPORT WE CAN OTHERWISE.

* CONSIDERABLE CARE MUST BE TAKEN TO MAKE SUCH A SYSTEM PALATABLE TO ENGINEERS:
  - > SELF-JUSTIFYING
  - > STEERABLE
  - > CREDIBLE

* A "SYSTEMS APPROACH" PROVIDES A POWERFUL WAY TO CREATE THE FRAMEWORK ("ARCHITECTURE") OF AN EXPERT SYSTEM:
  - > CACSD-III
  - > FOR ENGINEERING DESIGN IN GENERAL

* DEVELOPING 'SAMPLE TRANSACTIONS' IS ALSO A VALUABLE EXERCISE.

* DEVELOPING AN EXPERT SYSTEM CONCEPT REQUIRES A NEW LOOK AT THE PROBLEM --> VALUABLE INSIGHTS AND EVEN IMPROVEMENTS IN UNDERLYING "EXPERTISE" AND SOFTWARE.

CACSD-III-33

PLANS FOR THE FUTURE

1. FULL LISP IMPLEMENTATION - WITH NUMERICS & CAPABILITY TO RUN CONVENTIONAL CACSD SOFTWARE

2. MORE EFFORT IN AUTOMATIC DESIGN
   > POLE PLACEMENT
   > 'BRITISH SCHOOL' APPROACHES
   > MODERN CONTROL (ATHANS, QUADRAT)

3. DEVELOP A PROTOTYPE OR BENCHMARK "REAL SYSTEM" - WITH CAPABILITY TO SOLVE "REAL PROBLEMS"

CACSD-III-34